DOCUMENT RESUME

ED 107 741 UD 015 271

AUTHOR	Burnett, Jacquetta H., Ed.
TITLE	Computer Assisted Processing of Ethnographic Data. Volume 2 Part 1 of Final Report: Anthropological Study of Disability from Educational Problems of Puerto Rican Youths.
INSTITUTION	Illinois Univ., Urbana. Bureau of Educational Research.
SPONS AGENCY	Social and Rehabilitation Service (DHEW), Washington, D.C. Div. of Research and Demonstration Grants.
PUB DATE	72
NOTE	99p.
EDRS PRICE	MF-$0.76 HC-$4.43 PLUS POSTAGE
DESCRIPTORS	*Anthropology; Automatic Indexing; *Computer Programs; Data Analysis; Data Processing; Educational Research; *Electronic Data Processing; Ethnology; *Field Studies; Information Utilization; Input Output Devices; Research Methodology; Research Problems; Technological Advancement

ABSTRACT

The main purpose in developing the "package" of computer programs described in this guide is to use the automated capability of computers in the enormous job of handling anthropological field data. It is held that ethnography may be threatened with absolescence simply becasue so many man hours are involved in filing and sorting these types of data. One section discusses the procedures for getting the data from the field onto the computer in such a form that the researcher and computer can carry on from there with a minimum of clerk man-hour assistance. Another section introduces one to the program package. It gives an account of the requirements for using the programs on a computer installation and introduces important conventions which explain the reasons for many of the rules discussed in a subsequent section under key punching and copy editing. The next section discusses input-output routines and presents in some detail how one prepares jobs for the program package. The last five sections focus on various programs. Appendix A describes the procedure for transferring the data to tape for long term storage and/or as a duplicate copy. Appendixes B and C describes sample uses of the program package. A glossary of program package terms appears at the end of the guide. (Author/JM)

ED107741

COMPUTER ASSISTED PROCESSING[1]
OF
ETHNOGRAPHIC DATA

Volume II, Part I

of

Final Report

ANTHROPOLOGICAL STUDY OF DISABILITY FROM EDUCATIONAL PROBLEMS OF PUERTO RICAN YOUTHS

---

by

Jacquetta H. Burnett, Editor and Director

Contributors:

James Coulthurst, M.A.
Dale Good, Ph.D.
Roger Hunt, M.S.
Robert Long, B.A.
Henry Slotnick, Ph.D.

[1]This investigation was supported, in part, by Research Grant No. RD 2969 G 69 from the Division of Research and Demonstration Grants, Social and Rehabilitation Service, Department of Health, Education, and Welfare, Washington, D.C., 20201.

UD 015271

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

PREFACR

This guide is actually the product of the work and know-how of an undergraduate and graduate "computer" generation. The young men whose names appear on the title sheet are the authors of this guide. Burnett, the director and principal investigator was first student to their work and then editor of the volume.

The idea for developing some means of adapting the computer to assist ethnographers emerged during conversations between Burnett and Slotnick in 1968. Slotnick assembled the basic package, with the assistance of a colleague, Rodney Bussel.

An early edition of the guide was developed during 1968-69. This edition was used by Good, a graduate student in Educational Administration, and Hunt, an undergraduate in Engineering, who took over responsibility for handling the programs and their modifications as well as testing procedures for compilation and processing on data coming in from the field as we carried on the study of Puerto Ricans in Chicago. This involved continuing revision of the manual. During this period another undergraduate in Engineering replaced Hunt. By the end of the fieldwork phase of our Chicago study, the guide had been expanded and modified, all programs had been tested on our field data, and certain programs had been modified.

During the latter part of 1971, Coulthurst changed and modified certain of the accessing procedure, considerably reducing the knowledge of job control language needed to use the package for entering, manipulating, and processing data.

We continue to experiment with and develop the programs. In the future we hope to develop and add the EDIT program so that it can be used in copy editing data, in performing content analyses, and other analytic operations.

We gratefully acknowledge the financial assistance of the Social Rehabilitation Service, HEW, The Bureau of Educational Research and the College of Education of the University of Illinois. We would like to acknowledge with special thanks to the Measurement and Research Division of the Office of Instructional Resources for the loan of computer time and the willingness to give expert counsel, particularly to Richard F. Spencer, Director now deceased.

Those readers who are interested in obtaining copies of the source programs and object deck for BEDRES should contact Jacquetta H. Burnett, Bureau of Educational Research, University of Illinois, Urbana, Illinois.

TABLE OF CONTENTS

Page Number

Preface.....ii

Chapter

1. Introduction.....1

2. Processing Fieldnotes for Computer Entry.....6

3. The Bedres Program Package.....30

4. Bedres Input/Output Routines and Job Preparation.....38

5. Load Edit.....55

6. Edit.....57

7. Lister.....67

8. Index.....71

9. Concord.....74

Appendix

A. Transferring Files from Disk to Tape and Tape to Disk..80

B. Sample of Use of BEDRES.....83

C. Obtaining Stored Filenames and Cataloging.....85

Glossary.....87

LIST OF ILLUSTRATIONS

Figure		Page Number
1.	Flow Chart for Processing Fieldnotes	8
2.	Logsheet for Keeping Track of Fieldnote Entries	17
3.	Steps for Placing BEDRES on Disk	33
4.	Steps Followed in Using the BEDRES Package	34
5.	Overlay Structure for the BEDRES Package	36
6.	ID Card Preparation	44
7.	SETUP Card Preparation	49

1. INTRODUCTION

1.1. Purpose of Computer Assisted Processing of Ethnographic Data and the Guide.

The main purpose in developing the BEDRES "package" of computer programs described in this guide is to enlist the automated, clerical facilities of the computer to perform indices, concordances and limited analyses of data recorded in natural language (in our case in both English and Spanish). Part of our time has been spent in developing efficient, workable procedures for preparing the data to be "loaded" onto the computer. In addition, many months were spent, not only in creating (both through borrowing and inventing) programs that would file and sort data, but in working the "bugs" out of the initial programs so that the final programs worked precisely and reliably. This phase of the computer use-development took an unanticipated two years. During the final phase we have reduced to a minimum the knowledge of job control language needed by a user of the system.

We have accomplished the first objective in the plan for computer assisted processing of ethnographic data (CAPED): A researcher alone, with perhaps a part-time data processing clerk, the BEDRES Program deck, this manual, and a few hours with a programmer each week, can learn to continue to rework the data in the direction of recoding, standardizing, and systematizing it on theoretically significant grounds.

Most important, the initial compilation by the computer retains the open-ended, "serendipituous" character of the ethnographic field data, in which observations first are coded, so to speak, in a natural language

system, for example, English or Spanish. This feature of our CAPED design, seemed particularly special because most of the use of computers in anthropology has involved "automatic" classification, or pre-codification, prior to its being loaded on the computer.* We have moved computer loading back a step, and have provided a means of entering data "coded" in natural language as it has traditionally been done in ethnographic field notes· the second and third levels of abstraction, through recoding, are done not in the field, but by the researcher on the computer with the clerical aid of the computer. In other words, the same ethnographic data can be reworked numerous times according to several different possible theoretical schema, without high additional investment of man-hours of work, after initial processing and loading onto the computer.

In the simplest terms and at this stage it has been our purpose to enlist the computer's automated help in the enormous clerical job of processing anthropological field data. Some commentators frankly are impatient with, even contemptuous of, the mere clerical use of computers, in filing and sorting. They press on with the desire to use this sophisticated equipment to assist in systematization of theoretical significance. While one can appreciate the concern to link formulations familiar to anthropologists with formulations involved in sophisticated computer processing so as to facilitate significant data reduction, automated clerical help in processing ethnographic type data can be of enormous value to the employment of that data in systematizing anthropological inquiry.

After my own experience with processing ethnographic data in two different field studies, I became increasingly doubtful that natural history accounts of human behavior, which is what ethnographies are, could survive

*See Dell Hymes, ed., The Use of Computers in Anthropology, Mouton & Co., 1965.

as a research procedure in a high labor-cost economy. Ethnography could become obsolete simply because so many man hours are involved in filing and sorting this type of data. Moreover, often there is a very low ratio of the amount of information retrieved (not necessarily the significance of the information) to each man-hour of work on ethnographic field data. Perhaps this set of programs will help carry ethnography into the era of automation.

1.2 Contents of Guide

Compilation of the data, filing, and sorting, involve programs that index and perform concordances of the data. In the BEDRES program package, these "accounting services" are performed by INDEX and CØNCØRD. Indices are defined in this package as alphabetized listings of all the different words appearing in the data and the indication of the sentence numbers and files in which they appear. The concordance program utilizes the indices to list and print-out, on command, those sentences in the data which contain specific words or constellations of words, as well as a certain number of sentences preceding and following the target sentence.

To obtain an index however, BEDRES employs two programs: LISTER, which produces a list of all the words in a text: INDEX, which then, locates each word (or phrase) by the number of the numbered sentences(s) in the text where it occurs. CØNCØRD can then be used to "retrieve" and print-out the sentence context, or paragraph, in which a word or phrase appears. This procedure aides retrieval of meaningful, or context defined, information.

For example, if we wish to obtain certain information about a given individual, we can use LISTER, INDEX, and CØNCØRD in concert to locate that

information. LISTER and INDEX give one all the locations of mentions of a certain person. CØNCØRD, then narrows down the search by giving a bit of each context in which the person's name appears. With these hints one can narrow the search by setting up a priority list of the remaining most likely locations of the information wanted. Then through ordering a print-out of the full file, for each of the most likely locations, in order of priority, the researcher can locate the correct context.

EDIT is initially used to load data on to the computer from cards. EDIT acts as a filing system for data as well, since it requires that all data be labeled by date and type of source (fieldnotes or interview). Each set of data is also assigned unique codeword so that any operations performed on a dataset will not affect other data on the computer.

EDIT is a utility routine that is of major significance to the further analysis of ethnographic data once it is copied onto the computer, and indices and concordances have been done. We intended to use EDIT to help us with copy editing the text of the fieldnotes that was on the computer disk. EDIT, as now written, is too cumbersone for this use. EDIT, however, is used to separate portions of data from other portions this, allows the research to retrieve selected information pertaining to one person, one event, or one category. Most importantly, because EDIT can change individual words, phrases, and sentences in a file, it can be used to recode, or to standardize, descriptive accounts of items or events which conceptually are classed as the same. This is an important analytic step beyond mere filing, sorting, and locating.

Finally, EDIT is used to concatenate files, such that several small files can be put together in one large file. It is used to reorder

datasets into different relationships to one another. We use EDIT to put all our datasets into chronological order. Thus, EDIT is the star program in moving CAPED beyond the compilation stage.

The following chapters describe not only the programs of BEDRES, but also in Chapter 2, the procedures for getting the data from the field onto the computer in such a form that the researcher and computer can carry on from there, with a minimum of clerk man-hour assistance. (Indeed, remember the computer is also a typist of its stored texts.) Chapter 3 introduces the reader to the BEDRES Program package. It gives an account of the requirements for using the program on a computer installation and introduces important conventions which explain the reasons for many of the rules discussed in a subsequent chapter under keypunching and copy editing. Chapter 4 discusses input and output routines, and presents in some detail how one prepares jobs for BEDRES. Chapters 5 and 6 discuss "Load EDIT" and EDIT. LISTER, INDEX, AND CØNCØRD are discussed in Chapters 7, 8, and 9. Appendix A describes the procedure for transferring the data to tape for long term storage and/or as a duplicate copy. Appendixes B and C describe sample uses of BEDRES. A Glossary of BEDRES terms appears at the end of the guide.

The initial sections of the guide from the "Introduction" through "Input/Output Routines and Job Preparation" will give the reader a picture of CAPED as we developed it for this project. The other sections are probably more easily read and understood as the individual user adapts the package to his own use and computer installation.

2. PROCESSING FIELDNOTES FOR COMPUTER ENTRY

2.1 Outline of Procedures

Developing efficient, workable procedures for processing ethnographic data so that it can be "loaded" on the computer was a major job in the development of our CAPED. Preparation procedures were contingent not only on the set of programs used to compile and organize data, but also on the type of user services available from our computer installation. For example, during the first year of the project, our installation offered a remote terminal and disk sharing service which we tried to adapt to our use. As matters developed, it became more efficient to punch and copy edit cards for loading onto the computer.

Creation and development of programs for a "package" involved hours of trial, debugging, retrial, and re-debugging. The trials involved "experiments" with samples of data. But we did not use samples of our field data, until the program, or programs in combination, had worked well on more standard types of data. The design for the final stages of processing for computer loading had to wait until we were sure the package would work, at least, through the compilation stages. In the meantime, as fieldwork produced data, we used a traditional man-hour system of filing and cross-filing in file cabinets. As the CAPED system became more efficient, more and more of these traditional, hand-operations were dropped, and we began to transcribe tapes and handwritten notes directly on to the cards by keypunching (i.e., we dropped out section 2.3.2).

The procedures described in this chapter include the "hand" operations, as well as the computer dependent operations. We have starred those steps

that could be dropped out once the programs were reliable and indices and concordances were being printed out by the computer.

In the first section of this chapter we outline and diagram the processing procedures: the rest of the chapter spells out these steps in detail. As a whole, the chapter covers all those steps followed after the fieldnotes are collected and labeled up to the point at which the BEDRES program takes over.

A flow chart for processing fieldnotes might be outlined as follows:

Phase I. Labeling fieldnotes received from the field.

Phase II. Duplicating for cross-filing.

a. Principal sample file
b. Author file
c. Date file (see Phase IV)

Phase III. Check off scheduled interviews completed.

Phase IV. Processing of date-file copy for computer storage.

a. Editing original text
b. Keypunching
c. Listing
d. Editing printout
e. Correcting cards
f. EDIT cards onto disk

Phase V. BEDRES Program processing of fieldnote data sets.

a. LISTER
b. INDEX
c. Concatenation
d. CØNCØRD

These steps are diagrammed in Figure 1, pp. 8-12

2.2 Coding Computer Files to Hold Fieldnotes

2.2.1. File all fieldnotes by "name" so they can be entered on the volume table of contents. The names of the

Figure 1. Flow chart for processing fieldnotes.

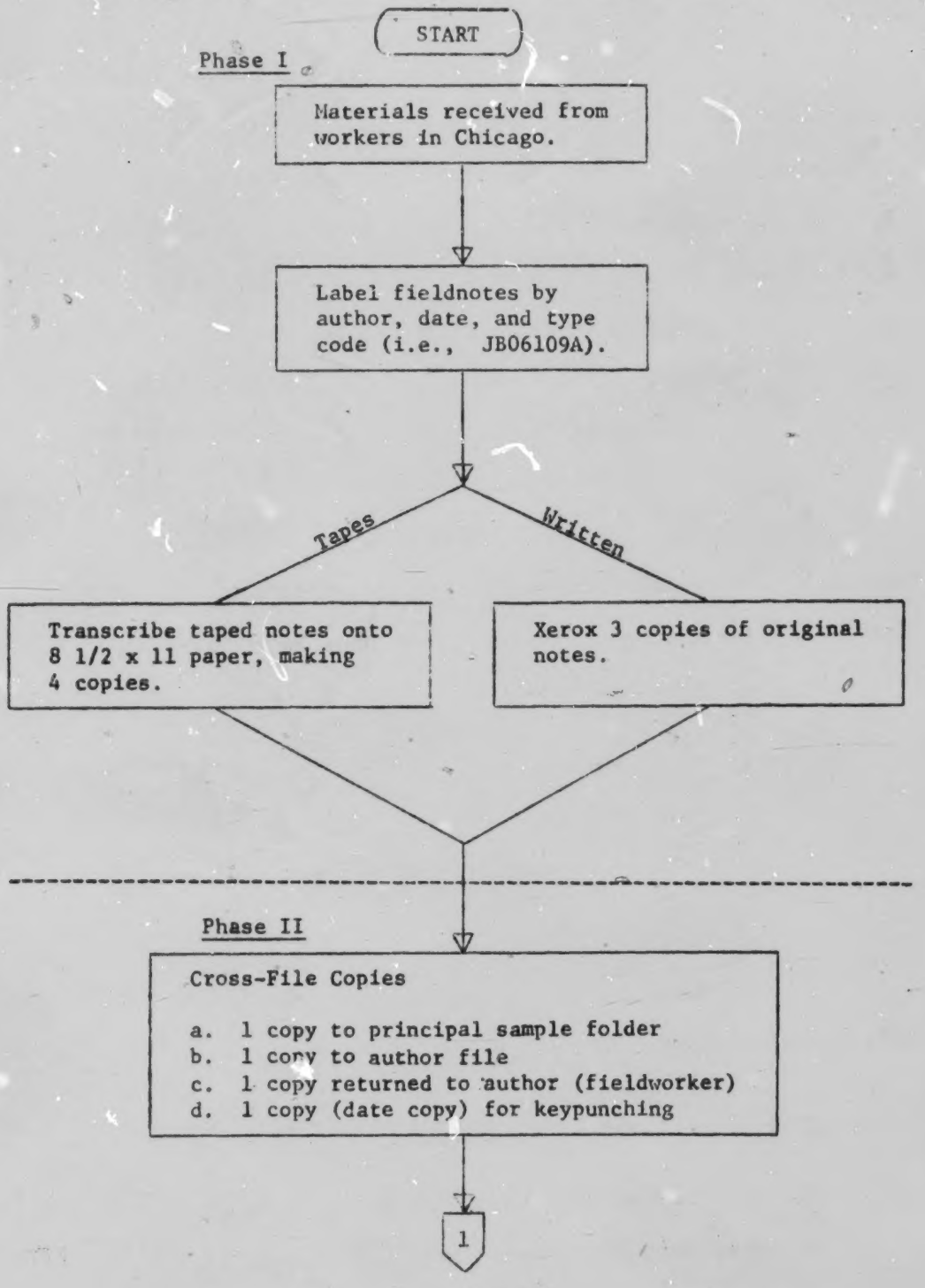

START
Phase I
Materials received from workers in Chicago.
Label fieldnotes by author, date, and type code (i.e., JB06109A).
Tapes
Written
Transcribe taped notes onto 8 1/2 x 11 paper, making 4 copies.
Xerox 3 copies of original notes.
Phase II
Cross-File Copies
a. 1 copy to principal sample folder
b. 1 copy to author file
c. 1 copy returned to author (fieldworker)
d. 1 copy (date copy) for keypunching
1

Figure 1, Cont'd.

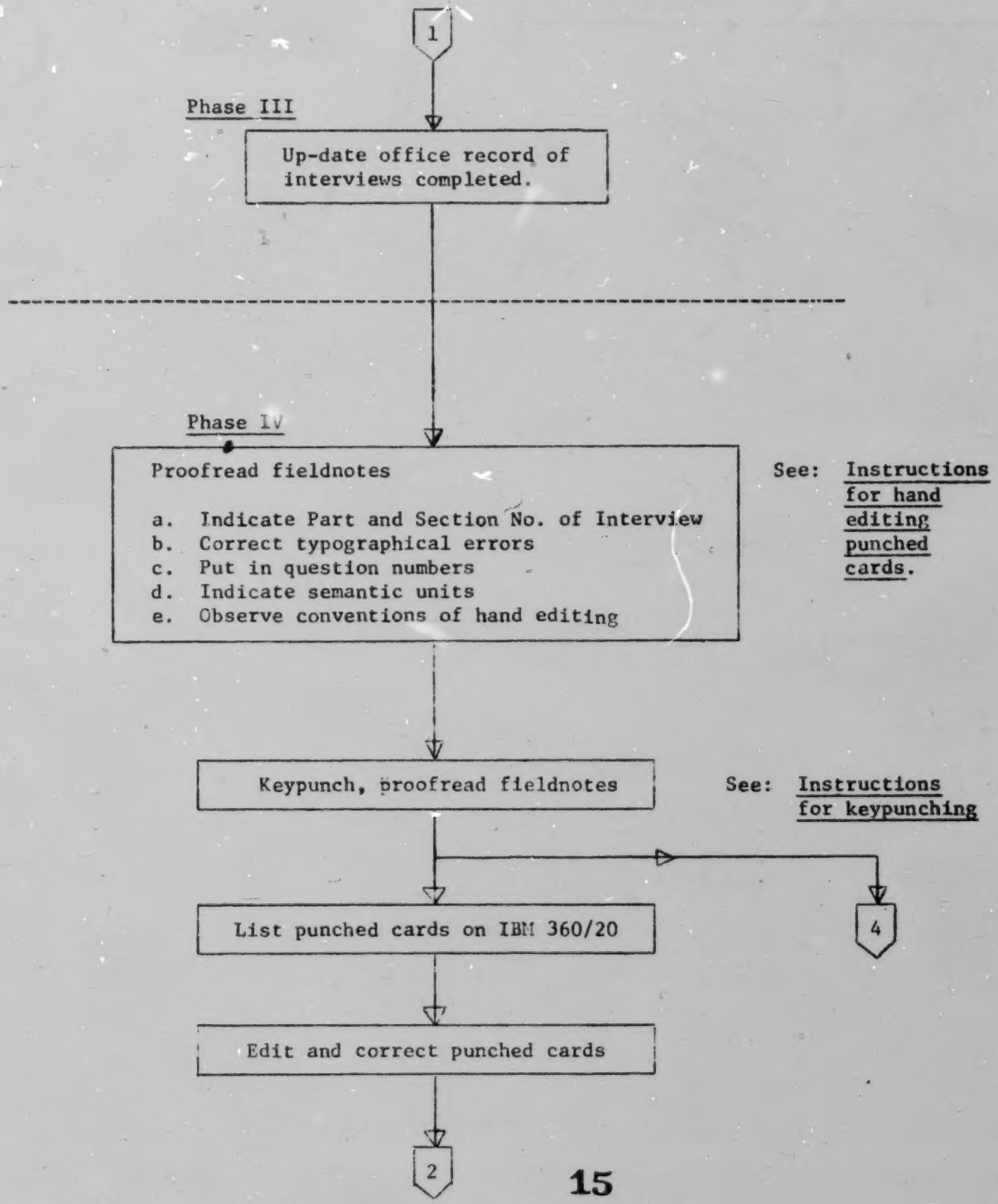


Figure 1, Cont d.

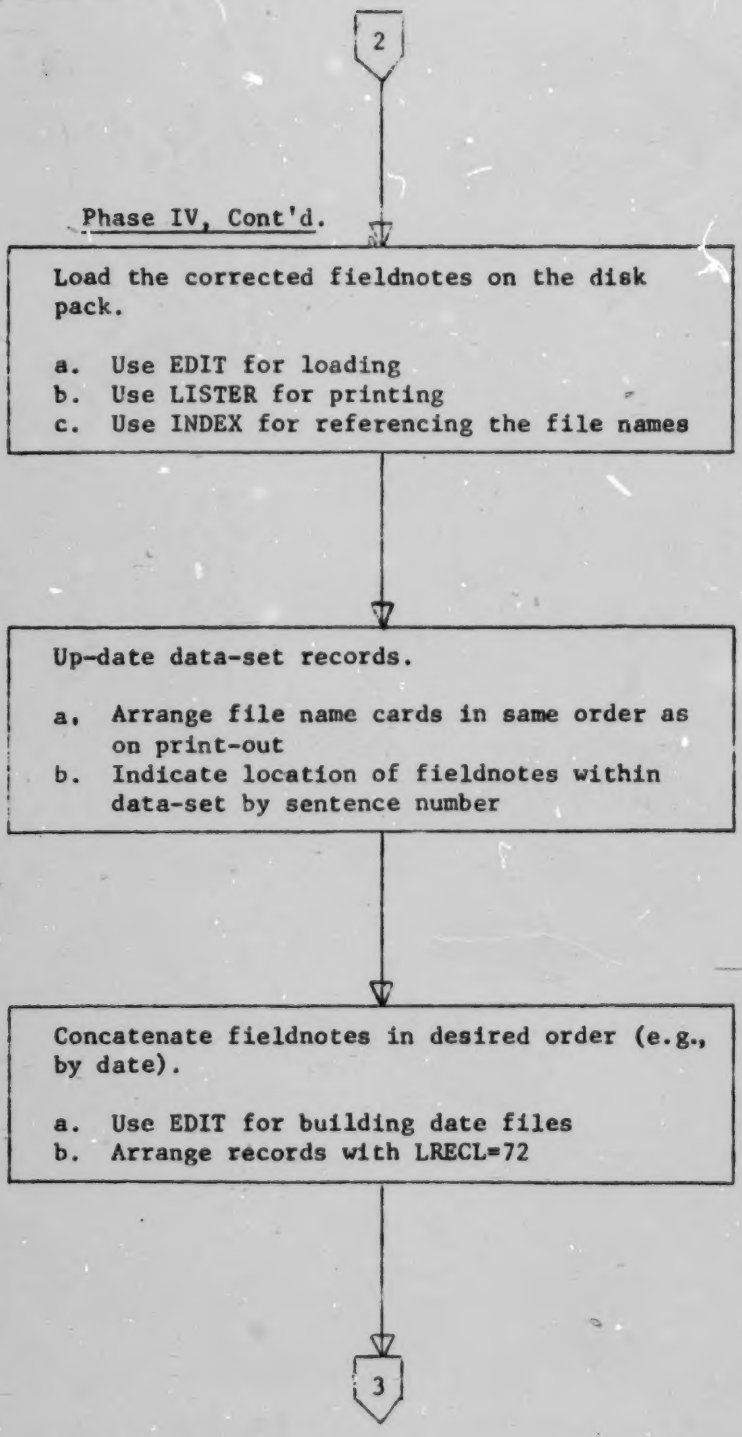

2
Phase IV, Cont'd.
Load the corrected fieldnotes on the disk pack.
a. Use EDIT for loading
b. Use LISTER for printing
c. Use INDEX for referencing the file names
Up-date data-set records.
a. Arrange file name cards in same order as on print-out
b. Indicate location of fieldnotes within data-set by sentence number
Concatenate fieldnotes in desired order (e.g., by date).
a. Use EDIT for building date files
b. Arrange records with LRECL=72
3

Figure 1, Cont'd

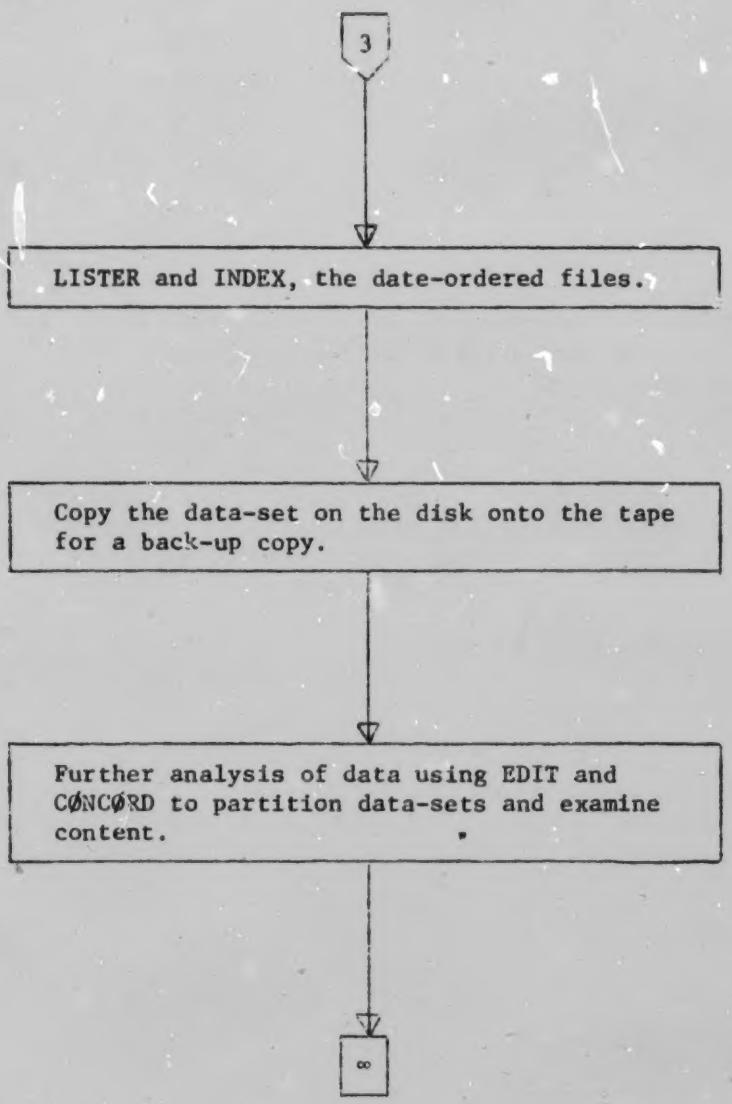
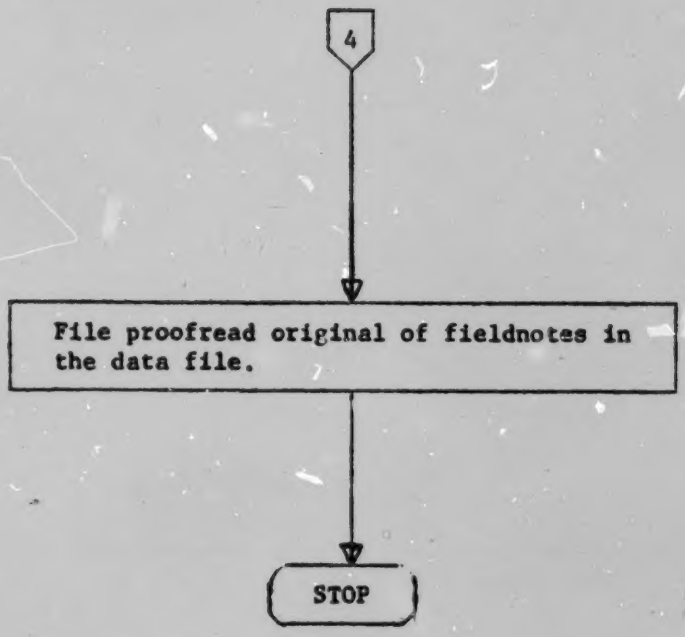

3
LISTER and INDEX, the date-ordered files.
Copy the data-set on the disk onto the tape for a back-up copy.
Further analysis of data using EDIT and CØNCØRD to partition data-sets and examine content.
∞

Figure 1, Cont'd.



files used to put field notes and interviews on the computer must:

a. Begin with an alphabetic character (A-Z) and
b. Be no longer than eight characters.

2.2.2. The name of a file will reflect:

a. The interviewer's initials, first and last name only,
b. The date of the transaction, and
c. The type of data that is to be input.

2.2.3. Using a set of eight characters to name each file, characters 1 and 2 should consist of the intials of the first and last name of the staff person who has written the data set.

Characters 3, 4, 5, 6, 7, represent the transaction; i.e., characters 3 and 4 are the month of the interview varying from January (01) to December (12), i.e., characters 5 and 6 are the day of the interview (01-31); and the 7th character is the numeric character of the year (0-9), but for the purpose of our research project 1969 is represented as 9, and 1970 is 0.

Character 8 (or the last character) is an alphabetic code indicating the type of data being treated:

Fieldnotes = (A-L): These include narrative accounts by fieldworkers of descriptions of events, extemporaneous interviews, or the account of the course of their day.

Interview transcripts = (M-Z): Are the accounts of the scheduled interviews with the principle sample youths and their social networks.

2.2.4. Each file can take up 999 lines or less on the computer. Therefore, if a set of data will run more than 999 lines you should start another file with the same basic name and *change only the last character of the name to the next* alphabetic character.

2.2.5. An example of a file name for fieldnotes is:

JB10208A

This file contains data from Jacquetta Burnett (JB) that was written in October (10) on the 20th day (20) in the year 1968 (8). The file contains written material of the fieldnotes type (A). If this file gets filled (i.e., runs to 999 lines) before the end of the fieldnotes, another file should be opened that is named:

JB10208B

Where B at the end indicates the second file for Burnett's fieldnotes on the 20th of October 1968.

2.2.6. An example of a file name for interview data is:

SW02059M

This file contains data from Sally Wolfe's (SW) work in February (02) on the 5th day (05) of that month during the year 1969 (9). The file contains data of the interview variety (M). If the 999th line of the file was filled and there were still more data to come you would open a new file named:

SW02059N

Where N shows that this file is the second one for Wolfe's work on that particular date.

2.2.7. If a field staff person generates data from more than one person or more than one interview schedule you should open the next logical file for each new set of data.

2.2.8. In 6 above no spaces should be left in the name of the file. Remember that the computer interprets a blank space as a character.

2.2.9. Write out completely words that are commonly abbreviated, such as street, avenue, company, doctor, etc. Abbreviations for Mr. and Mrs. may be used, but without the period (Mr & Mrs). The abbreviation U.S.A. may be used, but without periods and without spaces between the letters (USA). (US, however, may not be used because of possible ambiguity.)

2.2.10. The first appearance of a semantic unit, a set of two or more words that must be used together to convey a unitary meaning, should be enclosed thus: <Mr. Gonzalez>. For each file of up to 999 line, it is not necessary to continue this procedure after the first appearance of each individual unit however; but each <u>different</u> unit will have to be bracketed on first appearance in each file.

2.3 <u>Preparing Fieldnotes and Interviews for Computer</u>

2.3.1. The tapes are logged in by four different procedures.

1.1. They receive a file number (one person is assigned to do this).

Example: SW09220A

SW = Author's initials

09 = month

22 = day of the month

0 = last digit of the year, i.e., the date the notes were written.

The "A" designated that the notes are "Fieldnotes," and if it were an "M" it would designate the notes as "Interview Notes." (A - L = Fieldnotes; M - Z = Interview Notes.) So, the author is Sally Wolfe, the notes were written on September 22, 1970. If more than one interview is done in a day, then an "N," then an "O," and etc., is assigned to each subsequent interview.

.1.2. The file number is then put directly on the tape cassette with a sticky label. Then, the cassette is put on the top shelf of the file box marked "Taped Notes Received."

.3. The new file number (SW09220A) is typed onto "Copy-zip" paper and then entered on the large Log Sheet under "Taped Notes Received."
(See Figure 2 for example.)

.4. One person, so assigned, will keypunch an IBM card for that set of fieldnotes (SW09220A) and will put the card in a metal file drawer marked "Files Not Yet on the Disk" in order of person and date.

Figure 2. Logsheet for Keeping Track of Fieldnote Entries.[1]

Name	Taped Notes Received	Written Notes Received	Transcribed	Entered on Disk	Grouped [2]	EDIT	CØPY	INDEX	CØNCØRD
J. H. Burnett	JB03059M			JB03139A					
M. Cruz				MC05250N MC04200M MC06090P	MC10209A MC10209B				
S. Wolfe			SW08099M		SW10119N				
S. Pearlman/Llano		SP06150N		SP06150M	SP11149A				
L. Salces				LS01310A	LS02100A				
T. Herod			TH03060A		TH12199A				

1. The date file numbers or names are entered on "stickers" that are moveable.

2. If the user chooses to list several datasets together under a single filename, the data sets "grouped" under the filename are indicated in this column. Figure A.2. gives examples where two datasets are listed under a single filename.

2.3.2. The tapes are then transcribed. Four copies will result (one original and three carbons).

.2.1. The original goes to Burnett for editing. They are placed in the bottom shelf of the metal trays on the left side of her desk.

.2. One copy goes back to the author (usually sent to Chicago if that is where the author is located).

.3. One copy is filed in the Fieldnotes File under the author's name.

.4. The last copy goes in the principal sample file (in metal file cabinet). If, however, the fieldnotes make no mention of any of the Principal Sample, the last copy should be filed under miscellaneous.

2.3.3. After the notes are transcribed the person who transcribes them will transfer the file number sticker (SW09220A) on the Log Sheet from the "Taped Notes Received" column to the "Transcribed" column.

2.3.4. After Burnett edits the original copy, it is placed in the tray marked "Notes to be Entered on the Disk," third section from the top.

2.3.5. Entering data on disk: keypunch card method.

.5.1. Two cards are punched for each set of fieldnotes with the File Name of the notes. (See 2.2.1 - 2.2.6 above.)

.1.1. On the first card the file name begins with the first column (cc 1). *This card must be placed in the file drawer marked DATASET*

RECORDS to keep track of the table of contents of fieldnotes that have been sent to the computer.

.1.2. On the second card, indent five spaces and begin punching file name in the sixth space (cc 6). A period must be placed after the file name. This card becomes the first card of the deck.

.1.3. Regarding keypunching cards after the title cards, see "Instructions for Keypunching: 2.4."

.5.2. After the interview or fieldnotes have been punched, put the punched cards in the card file drawer under interview name. (Be sure the first card is put in the Dataset Records drawer.) Put the draft you punched from on the file shelf, "Copy to be Logged."

.5.3. To copy-edit your punched cards, take the punched cards to the Digital Computer Laboratory, and get printout on the IMB 360/20. Edit this according to instructions in "Instructions for Hand Editing Punched Cards." Then change your cards to correspond to the edited printout. THIS STEP MAY BE SUBSTITUTED FOR VERIFYING STEP.

.5.4. Put copy-edited cards in file drawer "Punched Fieldnotes to be put on disk" (BEDRES). Put edited printout in file shelf labeled "Correspondence" with note indicating that cards are ready for BEDRES.

2.3.8. Process for Written Notes Received

.8.1. Written notes are given a file name which is then placed on the "Log Sheet" under "Written Notes Received."

.2. The notes are then xeroxed (one original and 3 copies) and distributed exactly the same way as the typed transcribed notes.

.3. After Burnett edits the original copy it is placed in the tray (third from top) for entering on the Disk. ("Notes to be Entered on the Disk.")

.4. The notes are then entered on the Disk according to the directions in Chapter 6 under "Load Edit."

.5. After the notes are on the Disk, the file number (SW09220A) is transferred to the "Entered on Disk" column on the Log Sheet.

.6. Should the user decide to group two or more sets of fieldnotes under one file name, this should be indicated in the "grouped" column on the logsheet.

2.4. Instructions for Keypunching

2.4.1. Two cards are punched for each set of fieldnotes with the file name of the notes.

.1.1. The file name is made up of the interviewer's initials and the date that the interview or fieldnotes were made. (See 2.3. Preparing Fieldnotes and Interviews.)

.1.2. The first card:

.2.1. Begin the file name in card column one (cc 1).

.2. This card must be placed in the file drawer marked, DATASET RECORDS, to be used as a reference card for the interview or fieldnotes which are sent to the computer.

.1.3. The second card:

.3.1. Indent 5 spaces and begin the file name in the sixth space (cc 6).

.2. A period MUST be placed after the file name.

.3. This card must be the first card in the deck. It is the title card for the fieldnotes.

2.4.2. Rules for keypunching all other cards punched after the title.

.2.1. If you are punching an interview schedule on the card following the file name card you should punch the "Part" number, "Section" number, name of interviewer; followed by a period. Move to the next card to begin the text.

.2.2. Each new question or paragraph of the interview begins on a new card indented 5 spaces. The BEDRES programs will then make each question the first word of a new paragraph for easy reference. Also, for fieldnotes begin each new paragraph on a new card in column 6, for the same reason.

.2.3. Around each question number put in semantic unit indicators (<>). It helps to set the number off as a single word. The semantic unit indicators are also placed around special words or groups of words which taken together mean something different than

when used separately. The BEDRES programs create a list of these semantic units for easy reference.

Semantic unit indicators are only put around the words the first time they are used in each interview or set of fieldnotes (i.e., in each data set that begins with a file name). The BEDRES programs, through INDEX, automatically list all of the sentence numbers where the semantic unit is found even though the semantic unit indicators are only put around the semantic unit the first time it is used. The semantic unit should be two or more words long, since the INDEX program gives a listing, by sentence number, where all individual words may be found. Semantic unit indicators should be put around:

.3.1. Specific names of people including first and last names; nationalities, gangs, etc.

Example: <Louis Rodriguez>, <North American>, <Latin Kings>

.2. Schools

Example: <Von Humboldt>, <Lane Tech>.

.3. Word references to education; college trade school, primary, grammar, grade, etc.

Example: <double promoted>, <U of >, <auto mechanics>, <third grade>.

.3.4. Welfare offices

Example: <Northside Welfare Office>

.5. Street Corners

Example: <Birch and Maple>.

.6. Restaurants

Example: <Kelly's Chicken Inn>

.7. Foods

Example: <Rice and Beans>

.8. Idiomatic phrases

Example: <acid trip>, <running board>, <side swipe>, <real bummer>, <keeping cool>, <big hearted>.

.9. Names of Interview Schedules

Example: <Part IV, Section I>. <Parent Interview Schedule>. <Migration and Adjustment Questionnaire>.

2.4.4. The first integer of the question number that is followed by a period indicates the main question. Example: 15.0.0, 15 is the main question number. When there is a second number following it, the second number indicates a variation on or a subpart of the main question. The second number in turn ends with a period. If there is a third number following it, the third number indicates a variation on the variation or a subpart of the first subpart of the main question. The last number does not end with a period. Example: <15.2.1>. If there are no variations on the main question it may be written <15.0.0> or <15>.

2.4.5. Initials are used in the interview to indicate who is speaking. For every change in speaker begin a new card and indent to start in column 6 with the intials of the speaker. After the initials, add a colon, and skip two spaces before adding content.

2.4.6. Periods followed by a space always mean the end of a sentence or a thought. The file name card, every sentence, phrase or one word remark must end with a period. Without it the remarks that follow are added to the preceding sentence. On the other hand, there are no periods after abbreviations unless that abbreviation also ends the sentence.

2.4.7. Decimal points may be used in indicating money, etc. They do not act as periods since there is a character in the space immediately following it. Provided the period is followed immediately by some character it acts as a decimal point and not as the end of a sentence.

2.4.8. All reference to time should be changed to the 24 hour clock, thus eliminating the confusion of A.M. and P.M.
(i.e., 3:00 P.M. = 1500.)

2.4.9. Quotation marks at the end of a sentence must be followed by a period, question mark, or exclamation point. Example (".), ("?), ("!). If these were reversed to (.") etc., the BEDRES program would end the sentence with the period and add the quotation mark at the beginning of the next sentence.

2.4.10. Skip two spaces after a period before beginning the next sentence.

2.4.11. You may end a word in column 80 of a card, and begin a new word on column 1 of the next card. The computer will automatically skip a space.

2.4.12. If you cannot fit an entire word on a card you may put a hyphen at the end of the letters on the first card and continue the word starting on column one (CC1) of the next card. The BEDRES programs will make the word whole again on the printout. The word need not be divided into syllables since on the printout the word will be properly reconstituted.

2.4.13. If a word contains a hyphen but cannot fit on one card, put a hyphen to indicate the rest of the word is continued on the next card, and another hyphen on the next card to indicate the word is hyphenated.

2.4.14. If you make an error such as leaving out a word, type a new card and insert the word; whatever will not fit on the first card, put on another new card immediately following the corrected one. The BEDRES programs will not leave a large gap but will adjust the sentence to normal spacing.

2.4.15. After the interview or fieldnotes have been punched, put the punched cards in the card file drawer under the interviewer's name and put the original draft you punched from in the file shelf, "copy to be logged."

2.5. Instructions for Hand Editing Punched Cards*

2.5.1. Arrange the punched cards in chronological order by author.

2.5.2. Take the punched cards to the Digital Computer Laboratory and get a printout on the 360/20 (or listing of cards). Ask the 360/20 operator to begin each set of fieldnotes at the top of a new page. The hand editing can be done from 360/20 printout and then the cards must be changed to correspond to the edited printout before they are loaded onto the storage device.

2.5.3. All editing marks should be made in red pen to catch the eye. Circle incorrect spelling or punctuation and note above the error what change should be made.

2.5.4. The first line of the listing of punched cards should be one word, one sentence, paragraph, indicating the author of the interview or fieldnotes and the date it was made. This is the file name, and the title for those notes. Each file name should be indented 5 spaces, end with a period, and NOTHING ELSE SHOULD FOLLOW ON THAT LINE.

2.5.5. In fieldnotes that result from interview schedules each question answered must have a number. If numbers have not been indicated, check the loose-leaf notebook entitled "EPIC Questionnaires," find the corresponding interview schedule and decide which question is being answered and put the number by the question.

*The editing instructions that appear here have been covered in the previous section on keypunching. They are repeated here so that the relation of earlier directions to editing is clear and so that editorial directions are consolidated in one place, for purposes of instruction.

The question number should begin a new line and be indented 5 spaces. The first question number indicates the main question and is followed by a period when there is a second number following it. The second number indicates a variation on or a subpart of the main question. The second number in turn ends with a period if there is a third number following it. The third number indicates a variation on the variation or a subpart of the main question. The last number does not end with a period. Example <15.0.0> or <15>.

2.5.6. Semantic unit indicators

.6.1. Around each question number there must be a semantic unit indicator. They help to set the question off.

.6.2. They should also be placed around special words or groups of words which taken together mean something different than when used separately. Semantic unit indicators are only put around the words the first time they are used in each interview or set of fieldnotes, i.e., each dataset that requires a file name. The BEDRES program can automatically list all of the sentence numbers where the semantic unit is found even though the semantic unit indicators are only put around the semantic unit the first time it is used in each data set. The semantic unit should be 2 or more words long, since INDEX gives a listing, by sentence number, where all

individual words may be found. Semantic unit indicators should be put around:

(1) Specific names of people; nationalities, gangs, etc. (See the section on keypunching instruction for specific examples.)

(2) Schools

(3) Word references to education; college; trade school, primary, grammar, grade, etc.

(4) Welfare offices

(5) Street corners

(6) Restaurants

(7) Foods

(8) Idiomatic phrases

(9) Names of Interview Schedules:

<Part IV, Section IV>, <Formal Education Questionnaire>

2.5.7. Initials are usually used to indicate who is speaking. Every change in speaker should begin a new line indented 5 spaces, and followed by a colon. Two spaces should be skipped after the colon before content is added.

2.5.8. The first few lines of a dataset should indicate the interview part and section being given. It should be described with a number, and with the title for that interview schedule. If this has not been indicated, decide what interview schedule the questions are part of and find a blank interview schedule that covers those questions.

2.5.9. Correct all misspelled words.

2.5.10. Check for other punctuation conventions.

.10.1. Quotation marks at the end of a sentence must be followed by a period, question mark, or exclamation point. Examples (".) ("?) ("!). When this convention is reversed to (.") etc., the computer ends the sentence with the period and adds the quotation mark at the beginning of the next sentence.

.2. The title, every sentence, phrase or one word remark must end with a period. The period indicates the end of something. Without it the following remarks are added to it. There are no periods after abbreviations unless that abbreviation also ends the sentence.

.3. Decimal points may be used in indicating money and other figures where needed so long as the point never ends the number but is followed by some character. They do not act as periods since there is a character in the space immediately following it.

2.5.11. All references to time should be changed to the 24 hour clock, thus eliminating the confusion of A.M. and P.M.

2.5.12. Cross out portions that have been duplicated due to keypunching errors that were not removed before being printed out.

2.5.13. Edit cards to correspond to edited printout. Put the cards in the file drawer "punched fieldnotes to be put on disk" (BEDRES). Put the edited printout in file shelf "CORRESPONDENCE" with note indicating cards are ready for BEDRES.

3. THE BEDRES PROGRAM PACKAGE

3.1 Programming Language

The BEDRES Package programs are written in IBM Basic Assembler Language (BAL) for use on IBM System/360 computers*, though the user need not be familiar with assembler language programming. Because basic assembler language was used, the functioning of the programs is faster and more efficient than would be the case had some other programming language been used. Further, the functioning of the programs is independent of the operating system used by any given computer installation.

3.2. Package Conventions: Preparing Data for Entry

The package utilizes a minimum of conventions, some of which are discussed in the chapters for each of the programs. Those conventions which are relevant to keypunching and editing data, or to explain why certain editorial procedures must be followed.

Paragraphs are defined as being indented five spaces (five blank card columns). Periods followed by blanks or at the ends of lines are taken to indicate the ends of sentences. Thus, abbreviations and numbers cannot end with periods. Periods embedded within strings of characters (as ABC. DEF or 123.456) are not taken to indicate the end of a sentence. Commas cannot be used to separate thousands, for example, from hundreds: 10,000 is not recognized as ten thousand, but as "ten comma zero-zero-zero." Consequently, commas should not be used with numbers. Hyphenated words may appear at any point in a line or a card (i.e., a record) as long as the

*Models 50 and up.

Table 1. Legal Characters for the BEDRES Package.

Symbol		Type		
		Special Character	Punctuation	Alpha-Numeric
A				X
B				X
C				X
D				X
E				X
F				X
G				X
H				X
I				X
J				X
K				X
L				X
M				X
N				X
Ø				X
P				X
Q				X
R				X
S				X
T				X
U				X
V				X
W				X
X				X
Y				X
Z				X
0	Zero			X
1	One			X
2	Two			X
3	Three			X
4	Four			X
5	Five			X
6	Six			X
7	Seven			X
8	Eight			X
9	Nine			X
¢	Cent Sign			X
.	Period, Decimal Point	X	X	X
<	Less-Than Sign	X		
(	Left Parenthesis		X	
+	Plus Sign	X		
|	Logical ØR	X		
&	Ampersand, Logical + And	X		
!	Exclamation Point		X	
$	Dollar Sign	X		

Table 1. Legal Characters for the BEDRES Package. (Con't.)

Symbol		Type		
		Special Character	Punctu-ation	Alpha-Numeric
-	Minus Sign, Hyphen	X	X	
/	Slash	X		
,	Comma		X	
%	Percent Sign	X		
_	Underscore	X		
>	Greater-Than Sign	X		
?	Question Mark		X	
:	Colon		X	
#	Number Sign	X		
@	At Sign	X		
'	Prime, Apostrophe		X	
=	Equal Sign	X		
"	Quotation Mark		X	

Figure 3 Steps for Placing BEDRES on Disk.

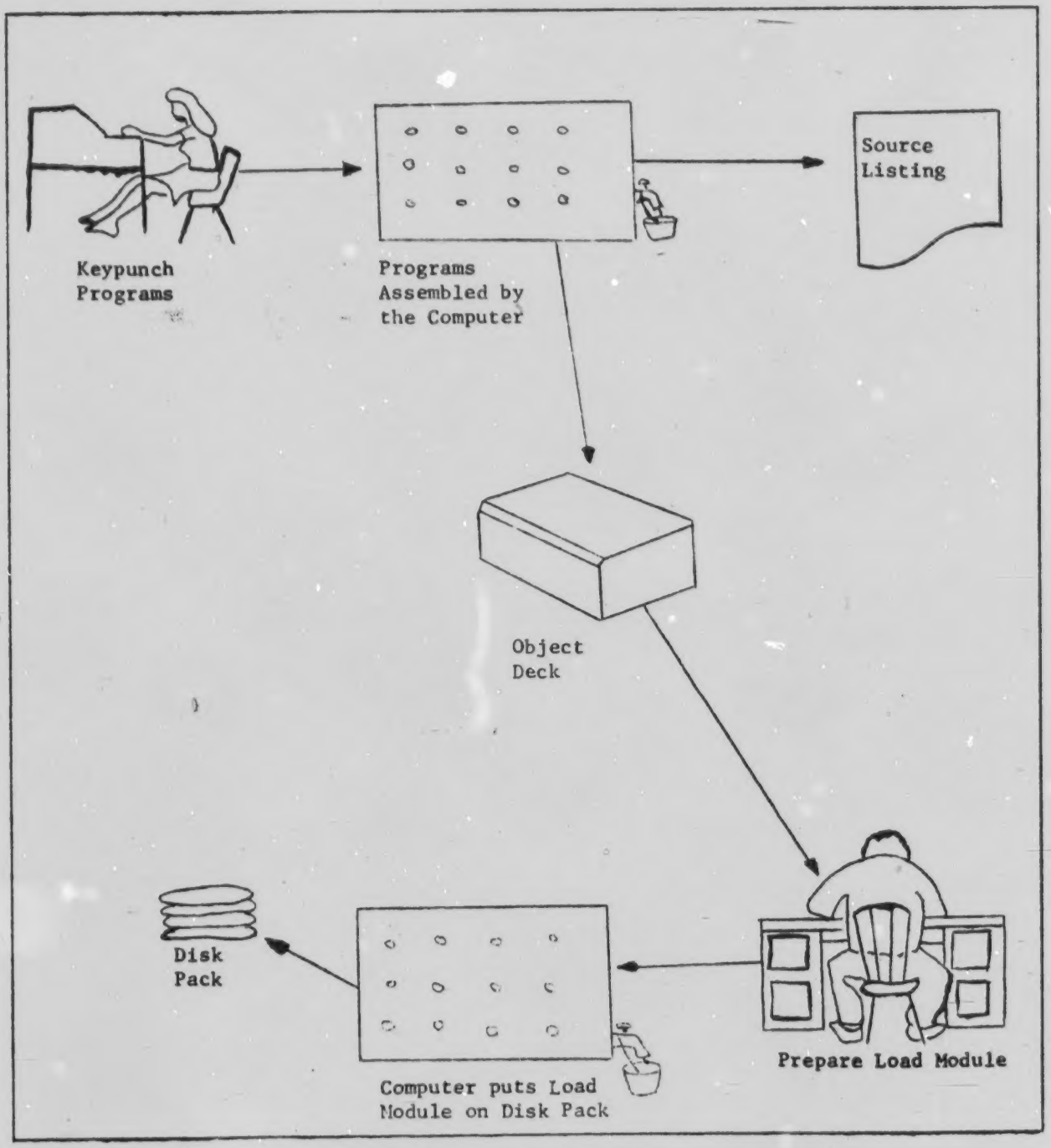
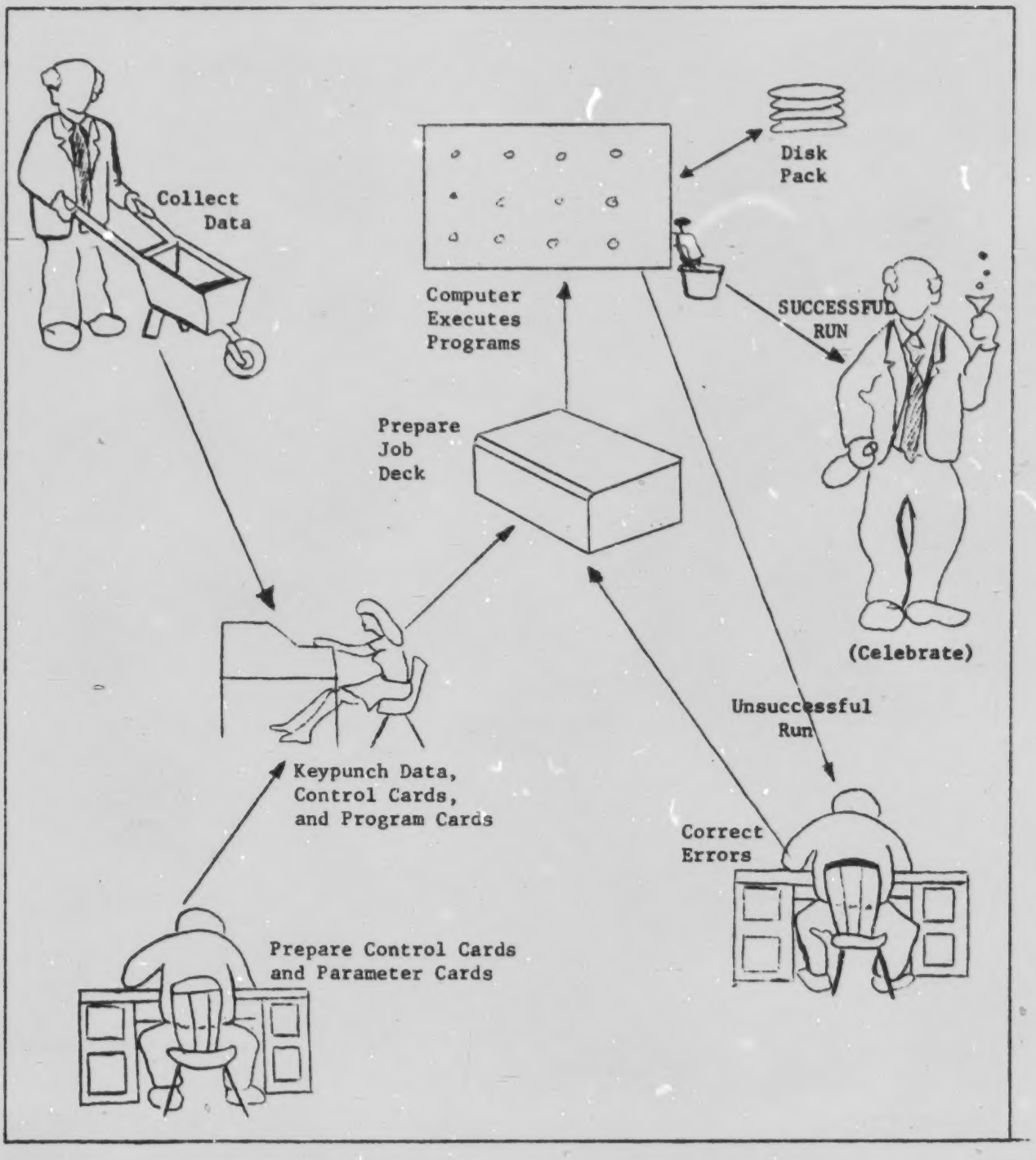


Figure 4 Steps followed in using the BEDRES Package.



hyphen is not the last character in the record. Hyphens appearing as the last character in a record indicate that the word preceding the hyphen is continued on the next record. If a word cannot be completed in the remaining space on the record, the hyphen can appear anywhere in the word, not necessarily between syllables. It, of course, must be the last nonblank character in the record. In addition to dividing words, hyphens are used to represent dashes. Dashes should be represented by two or more hyphens with no intervening spaces. Numbers following letters with no intervening spaces are taken as words. Thus "MAN1" and "MAN2" are both recognized as words.

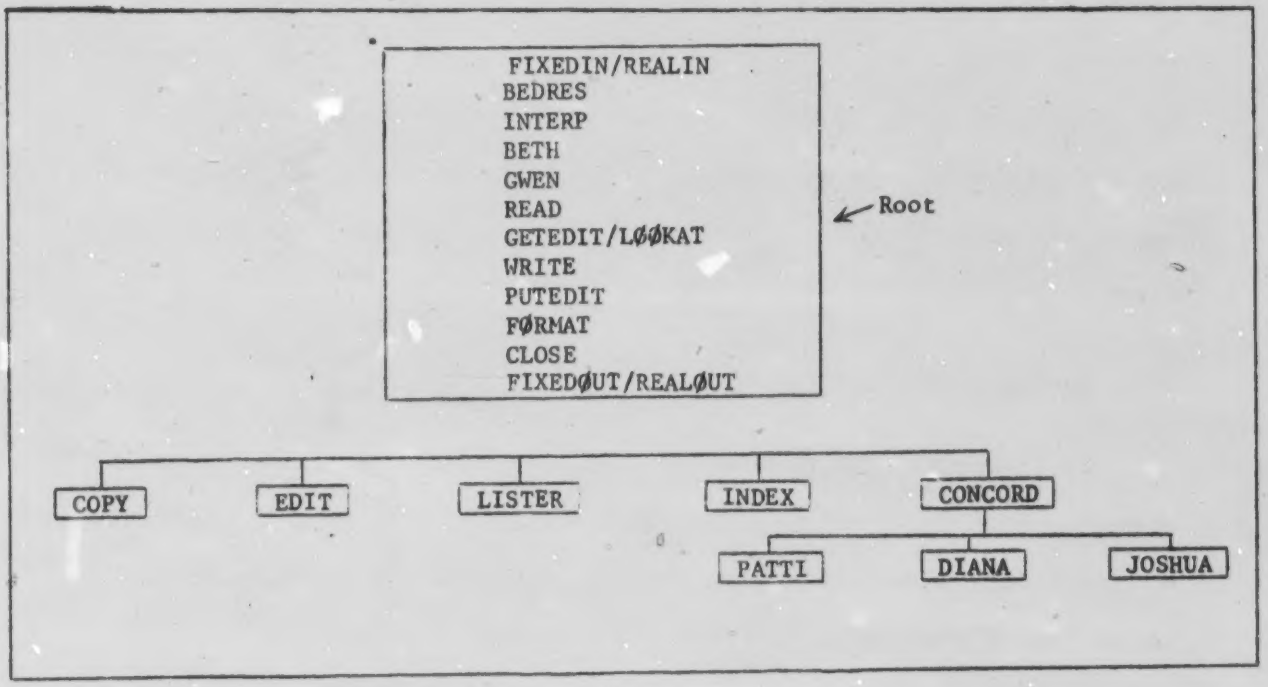
3.3. Execution of Programs

The programs in the package can be executed from source programs, object modules stored on a storage device such as a disk, object decks, or load modules. In this manual, it is assumed that the programs are to be run from a load module stored on a device such as a disk. The instructions for the preparation of a load module are found at the end of this chapter.

For storage the use of tapes and disks is strongly recommended; natural language data tends to be voluminous, and the use of a disk or tape will prove to be less expensive than the use of cards in both physical energy expenditure and input/output costs.

Figures 2 and 3 represent the steps usually followed in using the BEDRES Package. At the top of Figure 3 the source programs, original versions of the programs, are shown being keypunched and assembled by the computer. Essentially, the assembly "translates" the programs into code

Figure 5. Overlay structure for the BEDRES Package. (Boxed words are program names. The root section is utilized each time a branch program is executed. See the text of the following chapters for the functions performed by each program in this figure.



that can be executed by the computer. The assembly results in a source listing, a printed indication of the translation and, in this case, an object deck, cards containing the assembled program. The user should consult his computer installation to determine the procedure for assembling programs.

The object decks are then used to prepare a load module (assembled forms of the programs properly linked together). After the object decks and control cards are prepared, they are submitted to the computer which produces the load module and stores it on a device such as a disk pack.

In Figure 3 the data are being collected preparatory to being keypunched. Simultaneously, planning is taking place regarding how the data will be processed (bottom left). When the data has been collected and the planning completed, the data, control cards, and program cards are keypunched. (See Chapter 2 for keypunching instructions and Chapter 4 for control and program card descriptions). At the upper center of the page, the job deck has been prepared (see Chapter 4) using the data, control, and program cards. The job deck is then submitted to the computer (upper center). In the event of an unsuccessful run (bottom right), errors are corrected and the job deck is again prepared. The successful run (right center) is self-explanatory.

4. BEDRES INPUT/OUTPUT ROUTINES AND JOB PREPARATION

4.0 Input/Output Routines

BEDRES monitors all execution of the package program. It is the program that reads the program control card indicating which program the user wishes to execute. It also does preliminary scanning of the parameter boxes specified for the program. Parameter boxes indicate which files (data sets on the storage device) are to be used for input and output, and indicate which options, if any, are to be used in executing the program. The options are described in the chapters for each program.

4.1. Program Control Cards

Program control cards have the following format:*

**PROGRAM (Parameter Box 1) (Parameter Box 2) comments.

Two stars (**) appear in card columns (cc) one and two of all program control cards. The stars indicate to BEDRES that a program is to be called. Following the stars with no intervening spaces is the name of the program to be called. The name of the program replaces the word "PROGRAM" in the example immediately above. It is not necessary to spell out the entire name; only the first four characters are absolutely necessary.

Next come the parameter boxes--sets of left and right parentheses indicating the names of the data sets containing the data to be processed; the datasets on which the output is to be written; and the options, if any, to be used. The boxes are named symbolically to facilitate their use; in the INDEX routine, for example, the file containing the text dataset to

---

*A specific example of such a card might be:

**EDIT(INPUT:TEXT,CØL:1-72) (ØUTPUT:DATE,CØL:1-72,72) (CHANGE:*,CØL:1-80).

be analyzed is specified as:

(TEXT: DD name, CØL: i - j)

"TEXT" is the symbolic name of the box and the colon separates it from the rest of the information in the box. "DD name" would be replaced by the label of the file containing the text, and "COL: i - j" would indicate how many columns are to be read on the data set containing the text. "Col: i - j" can be interpreted as "the text is found in columns i through j." The letters "i" and "j" would, of course, be replaced by column numbers in an actual program run.

The parameter boxes for any given program do not have to appear in the order specified in the discussion of the programs in the following chapters. However, the order of the entries within each box MUST be as specified as shown.

The word "comments" following the last parameter box can be replaced by the user with information he would want printed out at the top of the program run. Note that the comments cannot include parentheses or periods (or they will be mistaken for parameter boxes). The period at the end of the program card is taken by BEDRES to indicate the end of the program card.

Program cards can extend over several cards though the total number of characters within the boxes, including blanks, cannot exceed 256. If multiple cards are used, only the first program card for each program can have the two stars in columns 1 and 2, and card columns 1 and 2 of continuing program cards can be used for any characters other than asterisks.

4.2. Multiple Program Execution

BEDRES can excute multiple programs sequentially: job steps can be stacked. For instance, a single run on the computer can execute a LISTER, an:: INDEX and a CONCORDANCE. The only requirement is that each program be called by its own program control card and that each program have its own job step.

4.3. Input from Cards

If the input to a program is from cards, the cards may be placed immediately after the prgram card, and their presence reported to the program by coding a single asterick (*) in the appropriate place in the parameter box for data for the program. For example:

**CØNCØRD (REQUESTS:*) (...) (...).

would evoke the CØNCØRD program and define the dataset called "REQUESTS" as being the cards following the program card. "REQUESTS" in the parameter box is the box's symbolic name. The first card following the dataset on cards should have "EØF" in cc 1 - 3. This card indicates the end of a dataset on cards.

Another way to use cards as input is to indicate a symbolic name for the cards in the appropriate parameter box and then define the dataset as being on cards using the IBM/360 Job Control Language convention. For example the program control card:

**CØNCØRD (REQUESTS:REQUESTS) (...) (...). indicates to the BEDRES monitoring program that the requests for this job step are to be found in the dataset labelled REQUESTS. Later in the same job

step the user should indicate:

```
//REQUESTS DD *
```

in Job Control Language. He should put the dataset on cards immediately after it and end it with /*.

4.4. Error Messages

BEDRES prints error messages when errors are detected in the specification of program cards or in the execution of package programs. These errors all cause the execution to be terminated. The messages, and the remediation to be taken in each case, follow immediately.

***ERRØR (BEDRES 1): NØ PRØGRAM CARD.

BEDRES could not find a card with two stars in cc 1-2. The probable cause is oversight or the absence of a **EØF card terminating the data cards for the previous job, if cards had been used for the program. Correct the error and resubmit the job.

***ERRØR (BEDRES 2): PRØGRAM NØT FØUND.

The program card called a nonexistent program, or the first four characters of the program name were misspelled. Correct the card and resubmit the job.

***ERRØR (BEDRES 3): MØRE THAN 10 BØXES OR 256 CHARACTERS FØR ØNE PRØGRAM CARD.

This message indicates that too many boxes or too many characters had been encountered for the program card. Check to make sure that boxes end with right parentheses and make sure the program card ends with a period.

***ERRØR (BEDRES 4): PRØCESSING STØPPED BECAUSE ØF ERRØR IN THE LAST PROGRAM.

This message indicates that an error had occurred in the program just run and BEDRES was terminating the run. When this message is printed out, an error message from the preceding program will also have been printed. Correct the error in the preceding program and resubmit the job.

4.5 Completed Execution

When BEDRES reaches the end of a job, it prints out:

ALL PRØCESSING HAS BEEN CØMPLETED.

4.6 INTERP

Note that BEDRES does not examine the contents of parameter boxes; that task is delegated to INTERP and the program called by the user. (See Figure 5, p. 37) Occasionally, a parameter box will not contain a colon (:) between the symbolic name and the information contained in the box. In this case, INTERP prints out the following message:

***ERRØR (INTERP): THERE IS NØ CØLØN AFTER THE KEY WØRD IN THE PARAMETER BØX.

In this case, the default option (see glossary) for that box is used. The error being considered can be corrected by placing a colon after the symbolic name in the appropriate parameter box.

4.7 Input/Øutput Routines

The input/output (I/Ø) routines are of no consequence to the ordinary user; with the exception of reading the input and printing the output, the user should see no indication of their functioning.

The following programs are included in the input/output routines: READ, WRITE, FØRMAT, GETEDIT, LØØKAT, PUTEDIT, FIXEDIN-REALIN, FIXEDØUT-REALØUT, AND CLØSE. (See Figure 5, p. 37) DATE is also included with these routines though it does not function directly in input/output.

4.8 Job Preparation

Preparation of a job is no more than the proper preparation and ordering of cards to input to the computer. It is assumed that the programs needed by the user have been keypunched and assembled and that a load module resides in the system (See Figure 3). The purpose of the rest of this chapter is to acquaint the user with the essential modules of instruction that are required for the BEDRES programs.

4.9 The /* ID card

An ID card contains billing information and information regarding the requirements of the job, e.g. an overestimate of the number of Input-Output requests to be used, and an overestimate of the amount of time required to process the data.

For example, the ID card:

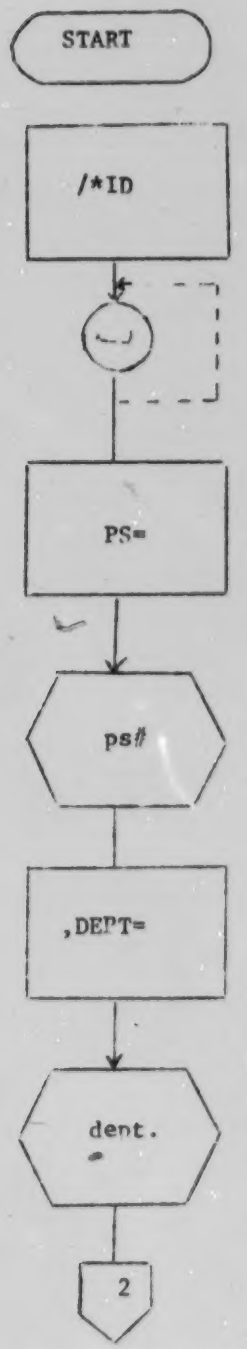
/*ID PS=555,CODE=MIKE,NAME=SMITH,DEPT=ED,LINES=15000,IØREQ=5000,TIME=2

indicates the following information.

The user's PS (problem specification) number is 5555. The charges for this job will be deducted from the account numbered 5555. The PS number is code-word protected, that is if someone were to try and use this PS# without specifying MIKE, the job would not run. The character string MIKE is replaced by XXXX on the print-out the user receives from the computer so that the print can be discarded without fear that some other user could retrieve it and run a job on PS#5555 if he didn't know the code word. A flow chart which specifies each of the parameters found on the ID card follows:

FIGURE 6 - ID CARD PREPARATION

ID Card



/*ID must appear in columns 1-4.

One or more blanks.

PROBLEM SPECIFICATION NUMBER

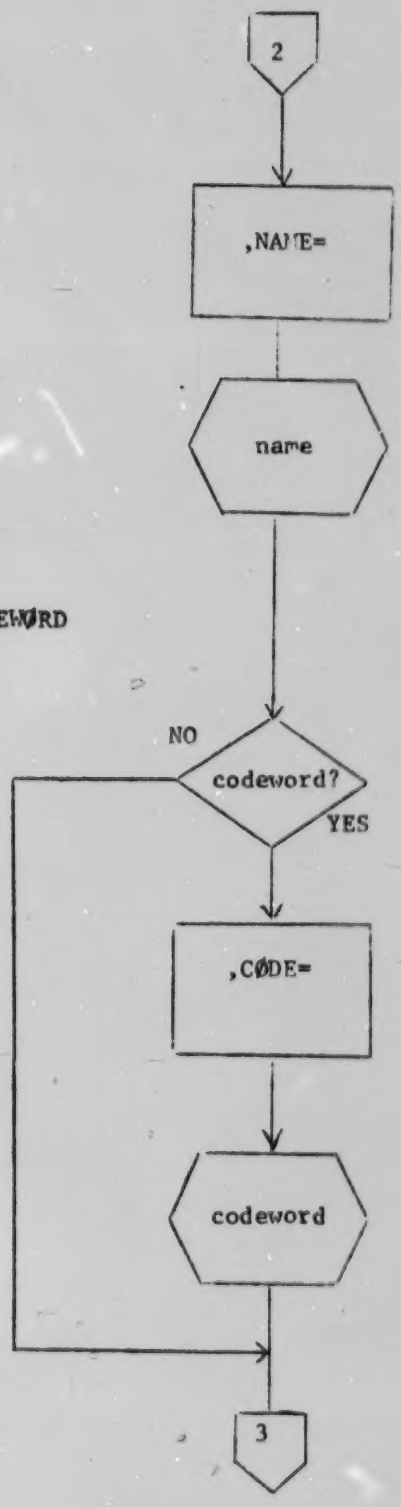
ps# is the user's four-digit Problem Specification Number

DEPARTMENT

dept is the department code associated with the ps#.

NAME

FIGURE 6 - Continued

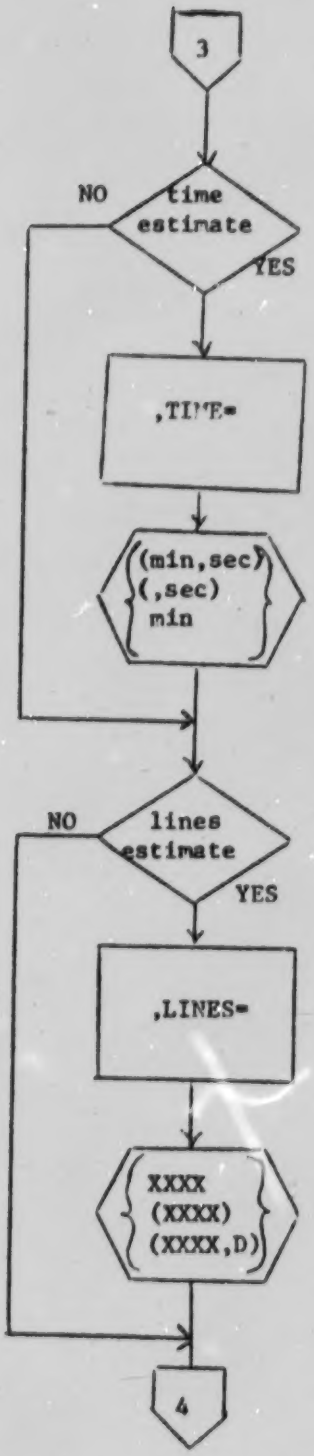


name is the user's name of 1 to 18 characters, including apostrophes if used. If name contains any characters other than A-Z and 0-9, then it must be enclosed in apostrophes; e.g., NAME='JØNES JØHN'. If an apostrophe is part of the name, it must be coded as two consecutive apostrophes; e.g., 'H.Ø''HARA'.

The user also has the option of having a code word assigned to his ps#.

code the code word associated with the ps#.

FIGURE 6 - Continued



TIME ØPTIØN

Do you wish to override the default time estimate of one minute of CPU time?

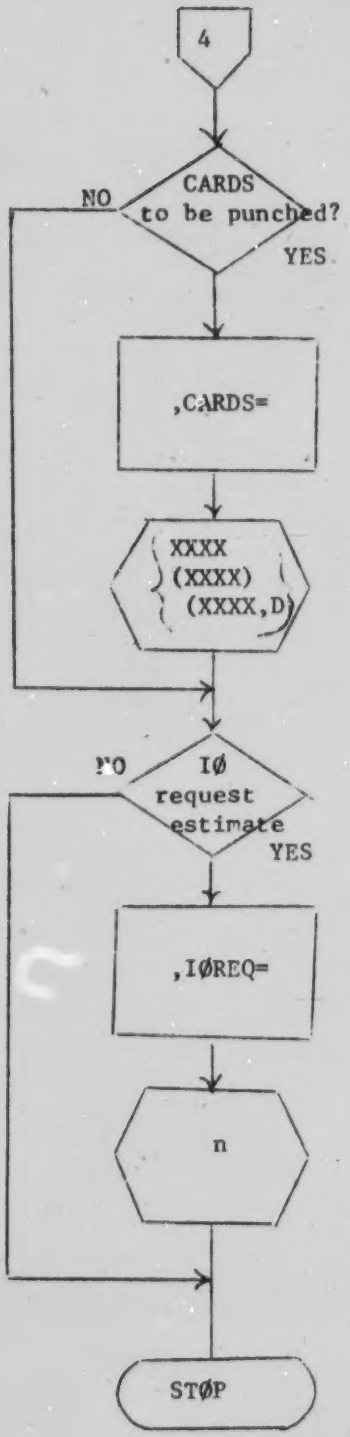
Min and sec must be integer constants.

LINES ØPTIØN

Do you wish to override the default estimate for printed lines of output which is 2000?

XXXX Lines may be printed; the D indicates that a dump is to be attempted if this limit is exceeded, and that the limit is to be ignored during printing of any Abend dump output.

FIGURE 6 - Continued



CARDS ØPTIØN

Do you wish to have cards punched? The default is zero.

XXXX is the number of cards that may be punched; the D specifies that a dump is to be attempted if this limit is exceeded.

IØREQ ØPTIØN

Do you wish to override the default of 1000 INPUT/OUTPUT requests?

Here, n is the number of INPUT/ØUTPUT requests. The program will Abend if this limit is exceeded.

End of the ID card for the BEDRES package.

4.10 The SETUP card

This card notifies the machine operator that the device mentioned should be mounted for the successful completion of the job. In our case the SETUP card:

/*SETUP UNIT=DISK, ID=DK0023

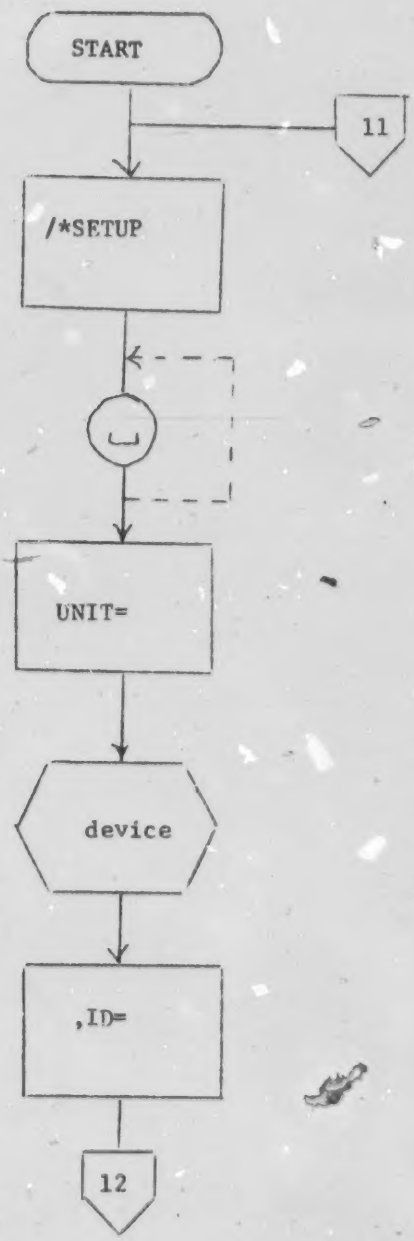
causes the disk numbered DK0023 to be mounted and accessed for the execution of the job. It will be remembered that for our project the program load modules and the datasets that contain the field notes are all on the disk DK0023. The appearance of a /*SETUP card causes a job to be placed in a setup-hold, and the setup information is written to the operator's console. When the operator has mounted all disks or tapes requested, the job is then released for execution. A flow chart indicating the parameter of the SETUP card follows:

FIGURE 7 - SETUP CARD PREPARATION

5.13.3. /*SETUP Card

Format:

/*SETUP UNIT=device,ID=(Ser#,options)

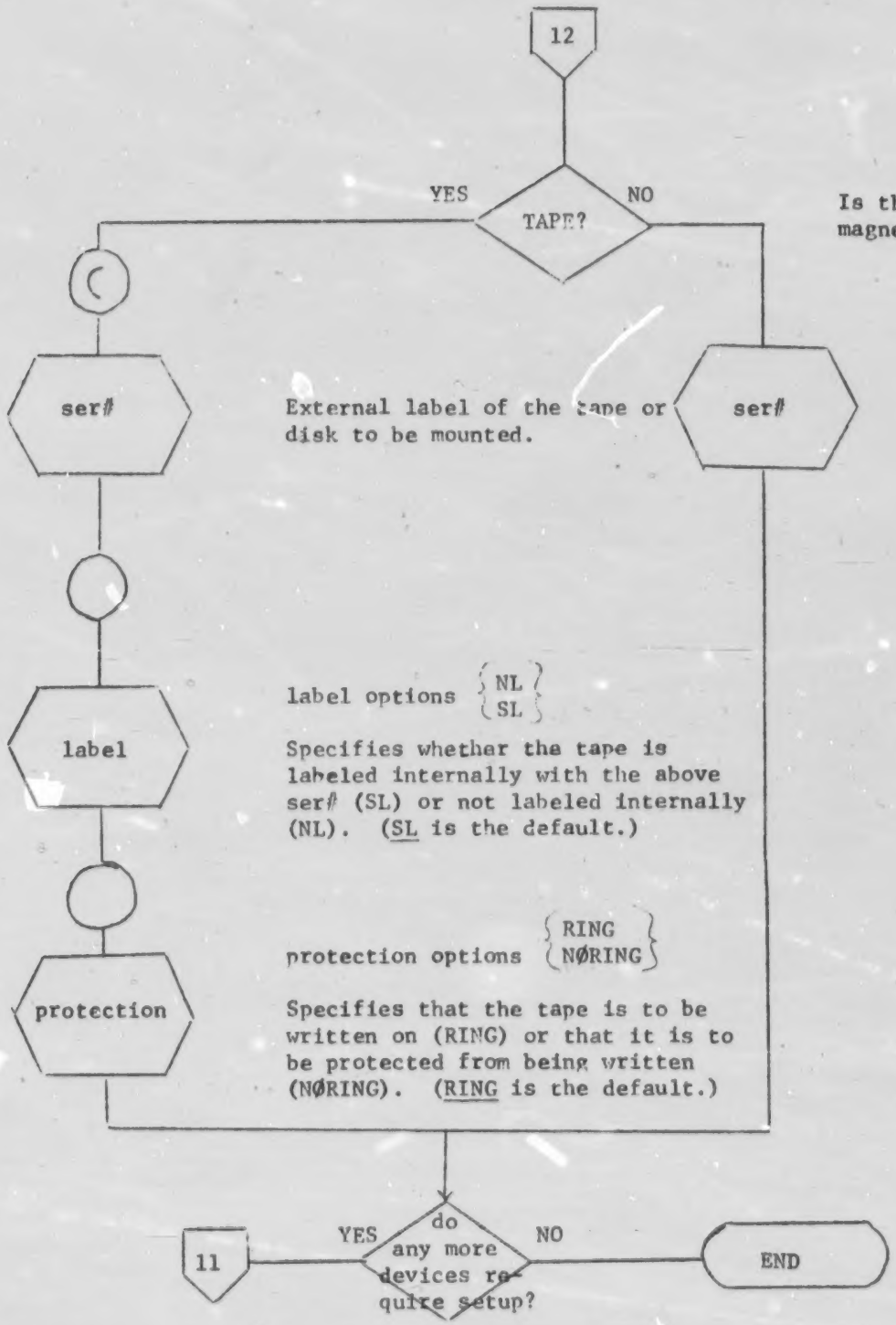


One or more blanks.

Device choices:

2314 - 2314 disk
TAPE - 9 track - 1600 BPI tape
TAPE8 - 9 track - 800 BPI tape
TAPE7 - 7 track tape

FIGURE 7 - Continued


12
YES
TAPE?
NO
Is the device a magnetic tape?
ser#
External label of the tape or disk to be mounted.
ser#
label
label options { NL SL }
Specifies whether the tape is labeled internally with the above ser# (SL) or not labeled internally (NL). (SL is the default.)
protection
protection options { RING NØRING }
Specifies that the tape is to be written on (RING) or that it is to be protected from being written (NØRING). (RING is the default.)
do any more devices require setup?
YES
NO
11
END

4.11 The // EXEC card

This card serves to begin a job step and to indicate to the operating system that a program which is a member of the load module called BEDRES is to be executed. It also contains code words for special requirements of each BEDRES program. These are specified in the sections dealing with specific programs such as INDEX or LISTER.

// EXEC PGM=BEDRES, WORDNAM=Blue, SENTNAM=Yellow

The user should remember that the EXEC card also signals the beginning of a different job step. For the BEDRES package usage it has been useful to put each separate program in its own job step. If, for example, the user wants to get a list of the contents of a dataset with field notes in it and an alphabetical list of the different words in the dataset he would write a program with two or more job steps. The first step might be called a "LISTER" step since he would call the LISTER program first. The second step could be called an "INDEX" step in the users mind since that step could call INDEX into execution. In BEDRES jargon then the user can create CØPY, EDIT, LISTER, INDEX and CØNCØRD steps in any legal sequence that he desires. A word of caution, however, the IBM 360/75 systems cannot handle jobs with a ridiculously high number of steps. As a rule of thumb the user should limit his jobs to around 10 steps and fewer.

4.12 The // CØNTRØL CARD

This card precedes the program control card (See section 4.1), for each BEDRES program. The card:

// CØNTRØL DD *

has only one entry, a star at least one space away from DD. The star indicates that the program control cards follow immediately. The first

card following the control card has to be a program control card. The user may put as many legal program control cards after the INPUT card as he desires. Emphasis is placed on "legal program control cards." There are two ways of defining legal at this point. First, it can be taken to mean proper keypunching. If the program control cards are mispunched either by pressing a key to punch a (.) instead of a (,); or if a package convention is violated in keypunching a card, then that card is illegal. Second, if a program control card calls a program that accesses an illegal file, the program card is not legal.

One of the hidden gremlins in the 360 operating systems is that the programs executing in a given job step can either read existing datasets or create datasets. It is not possible to create a dataset say WORDLIST with LISTER in a job step and expect that INDEX can read WORDLIST in the same job step. The remedy for this situation is to put a separate LISTER step and later a separate INDEX step in the job deck.

4.13 Program Control Cards--**Cards

The program control cards are instructions to the BEDRES programs. The card:

**EDIT(INPUT:TEXT,CØL:1-72) (ØUTPUT:DATE,CØL:1-72,72) (CHANGE:*,CØL:1-80).

calls the program EDIT to perform some operation. In the case of the card shown here the EDIT program is to perform its operation on the dataset defined as TEXT. Once the operations are finished the results are to be stored on the dataset defined as DATE. The specific changes to be performed are punched on cards that follow the **EDIT control card since a * is punched after CHANGE.

The card:

DELETE 86 TØ 1229.

defines the specific operations for EDIT to perform. In this example the user has indicated that EDIT should take the dataset with a DD name of TEXT in this step as input. EDIT was then directed to delete sentences number 86 through 1229 and put the remaining portion, sentences 1 through 85 on the output dataset with the DD name DATE.

The user should look back at the last parameter box on the **EDIT program control card. That parameter box is as follows:

(CHANGE:*,CØL:1-80).

The * after the word change indicates that the specific change instructions are on cards that begin immediately after the **EDIT card. When the user wants to put the changes on cards and include them in the INPUT file (defined as all of the cards between // cards DD * and /*) he should put a * after the change. The user should remember that at the end of the file on cards in INPUT there has to be a card that is coded.

**EØF

There is another way to tell the computer where to look for files on cards. We could have put:

(CHANGE:ALTER,CØL:1-80),

in the last parameter box in the example. That string of characters would instruct the BEDRES program to use the contents of the file with the DD name ALTER for instructions on the changes to be made. If this option is used and no other cards are put in the INPUT file then the user must define the file ALTER in the following way:

//ALTER DD *

(changes to be made)

/*.

The module of cards defining ALTER can go anywhere in the job step that it pertains to.

In the beginning the user will probably prefer to use the latter convention since it helps him to separate each of the datasets that need to be defined for execution into distinct modules of cards. Either of the two options work equally well, however.

4.14 The /* card

The card:

/*

must be put after the last card of any dataset that is on cards for the job step. Since // cards is a dataset that is generally on cards, a /* must be put at the end of the cards dataset. The larger files, such as datasets of field notes or word lists usually are on the disk pack.

5. LØAD EDIT

Although the BEDRES package contains a single program called EDIT, we single out one very important operation of that program, which is called LØAD EDIT for emphasis. "LØAD EDIT" is simply a special form of the EDIT program which enables the user to "load" data-punched on cards onto the disk in order to facilitate handling. It must be noted, however, that *this is the most critical step* in the entire BEDRES operation, since (1) the data are labeled for future reference at this time and (2) unique codewords are assigned to each dataset to insure that operations performed on any of the data do not interfere with the status of other data on the disk.

5.1. Procedure for executing LØAD EDIT

The following cards must be prepared for LØAD EDIT:

```
//  EXEC BEDRES,TEXT='(textname)',TEXTDSP='(NEW,KEEP)'
//  SETDISP='(NEW,KEEP)',WØRDNAM='(wordname)',
//  SENTNAM='(sentname)',REGIØN=100K
//CØNTRØL DD*
**EDIT(INPUT:CARD,CØL:1-80) (ØUTPUT:TEXT,CØL:1-80,80)
(CHANGE:*,CØL:1-80).
NONE.
/*
//CARDS DD *

[Data deck; i.e. data cards, follow at this position]

EØF
/*
```

The parameter box labeled *(textname)* indicates the name of the *dataset* to be entered onto the *disk*. This name is the *same* as the *filename* of the *dataset*, which consists of an eight character specification of the user, the date, and the type of data (See section 2.2.)

The parameter boxes labeled *(wordname)* and *(sentname)* are to be completed by using any word of seven characters or less in these boxes.

Each box must have a *different* word, however, and each new dataset will have its own *unique codewords*. To insure that the user does not misplace or lose track of which codewords apply to each dataset, it is wise to punch an *extra card* containing the codewords for the dataset and insert this card into the deck immediately *after* the *filename*. The example below will illustrate how this can be done.

5.2 Sample LØAD EDIT Program

```
/*ID PS=4545,NAME=BURNETT,CØDE=ANTHRØ
/*SETUP UNIT=DISK,ID=DK0023

//  EXEC BEDRES,TEXT='JB01049M',TEXTDST='(NEW,KEEP)',
//  SETDISP='(NEW,KEEP)',WØRDISP='(NEW,KEEP)',WØRDNAM='YELLØW',
//  SENTNAM='BLUE',REGIØN=100K
//CNTRØL DD *
**EDIT(INPUT:CARD,CØL:1-80) (ØUTPUT:TEXT,CØL:1-80,80)
(CHANGE:*,CØL:1-80).
NONE.
/*
//CARDS DD *

JB01049M

WORDNAM='YELLØW',SENTNAM='BLUE'

[Now place rest of data deck here]

EØF
/*
```

This sample program specifies that the dataset labeled *'JB01049M'* is to be loaded onto the *disk*, and that the codewords *'YELLOW'* and *'BLUE'* will be unique to this dataset for all future analysis on the computer.[1] Note that the first two cards in the data deck are (1) the *filename* of the dataset *('JB01049M')* and (2) the *special codewords* for this dataset.

[1] In each example which follows, the codewords *YELLOW* and *BLUE* are assigned to the dataset *JB01049M*. Individual users may apply any codewords they wish to datasets, as long as they are less than seven characters in length. See section 6.1.1.

6. EDIT

EDIT can be used to perform several other tasks. First, it can be used to move data from one record length (number of columns or characters per record) to another. For instance, it may be desirable to store data at 120 columns per record on a disk while the data has been keypunched at only 72 dolumns per record. The movement can also be from longer to shorter lengths. EDIT can be used to separate portions of data from other portions. For instance, reports of contact with three different informants may have been included on the same file, while the account from the middle informant may be the only one of interest. The account of what took place with the second informant can be separated from the other two through the use of EDIT. The third and last task that can be performed is the changing of individual words or sentences in a file. For example, data which were keypunched can be assumed to contain typographical errors. These errors can be corrected using the EDIT routine. In use, however, we have found the EDIT routine to be very cumbersome for doing very extensive copy editing. Instead, we now obtain a printout of our keypunched data cards and do the initial copy editing and correcting from that printout. Further development of the editor will be necessary before we can recommend its use for copy editing the texts on the computer, in addition to its other uses.

6.1. Procedure for Executing EDIT

EDIT is called by the following program card:

**EDIT(INPUT:inname, informat)(OUTPUT:outname, outformat, length)(CHANGE:chaname, chaformat).

The *INPUT* box contains two entries--the input filename (replacing *inname*), and the format for reading input file records. The input format replaces *informat* in the parameter box. In a similar manner, the name and format are specified for the edited material (in the *OUTPUT* box) and the file on which the changes to the input file are to be found (the *CHANGE* box). *Length* in the *OUTPUT* box indicates the number of *characters* output per line.

6.2. Default Options for EDIT

Default options exist for the formats and the *CHANGE* file. Unless otherwise specified by the user, the three formats read or write only the first 72 columns of each record. The *CHANGE* file default name is *INPUT* and assumes that the changes follow immediately after the program card. There are no other default *DD* names. The default length in the *OUTPUT* box is 72.

6.3. Using EDIT

EDIT can be used to concatenate small pieces of files on one large file. The concatenation procedure is useful for reordering field notes in different ways depending on how the researcher wants to view his data.

Operationally, to concatenate, EDIT is used to lift out a small piece of the corpus and to store it at the end of the file that the user is building. *The user should be aware of the fact that separate job steps are needed for each piece of corpus that is to be added to the large file.* Also, a single job submitted to the computer should have *no more than 10 job steps*. Otherwise, the machine will wait for

a long time until enough core is available and the job may terminate because the program was in the waiting state for too long a period of time, usually 8 minutes or longer.

If the user puts the following program cards in the deck for EDIT:

```
//CØNTRØL DD *
**EDIT(INPUT:JB01049M.CØL:1-80) (ØUTPUT;SCRATCH,CØL:1-80,80)
(CHANGE:*,CØL:1-80).
DELETE i TØ j.
EØF
/*
```

he is instructing BEDRES to scan the dataset called *INPUT* for *all* control instructions. Note the * after the word *CHANGE* in the last parameter box which, in BEDRES, means that the change instructions follow immediately on the cards.

If EDIT is to be used to make substantive changes to a file, it is necessary to know the number of each sentence to be changed. These numbers can be computed by hand, but that task can be done more easily by LISTER. If the *LISTING* option for *LISTER* is coded *YES*, a listing of the text will be produced with sentence numbers indicated.

6.4 Changes

Four types of changes, DELETE, ADD, MØDIFY, and NØNE, are possible using EDIT.

DELETE causes whole sentences to be dropped from a file. It is specified by:

DELETE i.

DELETE i TØ j.

or DELETE i, j.

The first example would cause sentence number i to be deleted while the second and third examples would cause sentences i through j inclusive to be dropped. The last two exapmles might be used to lift a given portion out of a file. For instance, assume that a user wished to generate an index on sentences 376 through 458 of a corpus which was 1375 sentences long. After specifying the *ØUTPUT* file in the appropriate box on the program card, he would indicate two deletions:

DELETE 1 TØ 375.

DELETE 459 TØ 1375.

After the deletions were executed, the *ØUTPUT* file would contain sentences to be inserted into the corpus. This type of change has the following format:

ADD BEFØRE i *' sentence '*.

ADD AFTER i *' sentence '*.

In the first example, the sentence(s) within the quotation marks (*' and '*) would be inserted into the *ØUTPUT* file immediately before sentence i. In the second example, the sentence would be inserted immediately after sentence i. Note that the quotation marks must be surrounded by blanks to assure that the apostrophes are not taken to be part of the sentence being inserted.

MØDIFY, allows strings of characters within sentences to be added or deleted. Note that the strings may be words, numbers, or punctuation marks. The format for this type of change is:

MØDIFY i CHANGE *' aa bb '* TØ *' aa cc '*.

MØDIFY i REPLACE *' aa bb '* WITH *' aa cc '*.

In both examples, the word *bb* in sentence "i" would be replaced by the word *cc*. Note that even though the word *aa* is deleted and then added again (resulting in no change) it serves a useful function, namely as an indication of where in sentence "i" the modification is to take place. Using such an *anchor word* is important in cases where the word to be deleted appears more than once in the sentence. *If no anchor word is used in such cases, the first appearance of the word in question will be modified.*

Note that words can be deleted from sentences by specifying

MØDIFY i REPLACE *' aa bb '* WITH *' aa '*.

MØDIFY i REPLACE *' bb '* WITH *' aa '*.

In the first example an anchor word is used, while the anchor word is eliminated in the second.

Words can be added to sentences through the use of

MØDIFY i REPLACE *' aa '* WITH *' aa bb '*.

Use of the MØDIFY change in this manner causes the word *bb* to be inserted in sentence i after the word *aa*.

Several MØDIFY changes can be made to the same sentence using the word *MØDIFY* only once if the terminal period is replaced by a comma and the *REPLACE...WITH...* portion is repeated. *The last character in the change must be a period.*

In making changes, the user should ascertain that the sentences being changed are specified in ascending order. (See error message *EDIT 16.*) Attempting to change a sentence that has already been passed through the computer will result in an error.

Specifying the change *NØNE* as a change will cause the entire file to be passed through unchanged. This change would be used in the event

a user simply wished to move the data on a file from one line length to another. For instance, data which was punched in the first 72 columns of a card can be moved to 120 characters per line (e.g., on a disk or tape) through the use of the change *NØNE*. Refer to the Example section under LØAD EDIT to see how this is done.

In the event that the sentence number is not specified, an error (***ERRØR (EDIT 5)* or *ILLEGAL CHARACTER IN INPUT RECØRD)* will occur. See the ****ERRØR (EDIT 5)* write-up in the following section for the appropriate remediation.

6.5. Error Messages for EDIT

***ERRØR (EDIT 1):
THE ØUTPUT LINE LENGTH AND FØRMAT HAVE BEEN INCØRRECTLY SPECIFIED.

This error message indicates that the last two entries in the ØUTPUT box have been incorrectly specified. Correct the error and resubmit the job.

***ERRØR (EDIT 2):
THERE IS NØT ENØUGH CØRE AVAILABLE FØR THE DATASTAK. SEE THE PRØGRAM WRITE-UP.

This error occurs when insufficient core (memory) is available to execute EDIT. This difficulty can be overcome by requesting more core (on computers using MFT or MVT). In other cases, see a consultant at the computer installation.

***ERRØR (EDIT 3):
THERE IS NØT ENØUGH CØRE AVAILABLE FØR THE CHNGSTAK. SEE THE PRØGRAM WRITE-UP.

See ****ERRØR (EDIT 2)* above.

***ERRØR (EDIT 4):
THE CHANGE DØES NØT BEGIN WITH "ADD", "DELETE", ØR MØDIFY".

As EDIT changes a file, it prints out each change specified by the user as that change is acted upon by the program. When this message appears, the change printed out immediately above it will be found to begin with an illegal word. Note that in specifying changes, only the first three characters of the key word need be specified. Thus DELETE and MØDIFY changes can be executed using *DEL* and *M/D*. This error message may have occurred because the first three letters of the key word had been misspelled.

***ERRØR (EDIT 5):
NØ SENTENCE NUMBER HAS BEEN SPECIFIED FØR THIS CHANGE.

This error occurs when the user has failed to indicate the number of the sentence to which the change refers. Indicate the sentence number and resubmit the job. Occasionally, this message will not appear and the message *INVALID CHARACTER IN INPUT STRING* will appear. Remediation is the same as specified above in ****ERRØR (EDIT 4)*.

***ERRØR (EDIT 6):
THE CHANGE SPECIFIED DØES NØT END; THERE IS NØ PEPIØD,
QUESTIØN MARK, ØR EXCLAMATION PØINT.

This message indicates that the change printed immediately above it on the print-out does not have a terminal punctuation mark. Insert a period, question mark, or exclamation point at the end of the change and resubmit the job.

***ERRØR (EDIT 7):
THE CHANGE DØES NØT INDICATE THE LAST SENTENCE TØ BE DELETED.

This message appears when a *DELETE* change, dropping more than

one sentence does not specify which sentence is the last to be dropped. Indicate the last sentence number and resubmit the job.

***ERRØR (EDIT 8):
THEPE IS NØ INDICATIØN ØF WHETHER THE ADDITIØN PRECEDES ØR FØLLØWS THE SENTENCE INDICATED.

In an ADD change, an error occurs if the words *BEFØRE* or *AFTER* do not appear. This error causes the immediately preceding message to be printed out.

***ERRØR (EDIT 9):
THERE IS NØ SENTENCE NUMBER INDICATING WHERE IN THE CØRPUS THE ADDITIØN IS TØ BE MADE.

The ADD change printed immediately above this message in the printout does not contain a sentence number. Indicate the number and resubmit the job.

***ERRØR (EDIT 10):
THERE IS NØ ØPENING QUØTATIØN MARK FØR EITHER THE INSERTIØN ØR THE DELETIØN.

If the words to be inserted or deleted in a MØDIFY change are not preceded by a '' surrounded by blanks, this message will be printed out. Insert the opening quotation marks and resubmit the job.

***ERRØR (EDIT 11):
THERE IS NØ INDICATIØN ØF WHAT IS TØ BE REPLACED.

In a MØDIFY change, nothing was found within the quotation marks following *CHANGE* or *REPLACE*. Correct the error and resubmit the job.

***ERRØR (EDIT 12):
THE PØRTIØN TØ BE REPLACED CANNØT BE FØUND.

The word or words to be dropped from the sentence specified in a MØDIFY change cannot be found in the sentence indicated in the

change. This error may have occurred because the sentence number was incorrectly specified or the portion to be deleted was incorrectly indicated. Correct the error and resubmit the job.

***ERRØR (EDIT 13):
THE PØRTIØN TØ BE INSEPTED CANNØT BE FØUND.

In a MØDIFY change, failure to indicate at least one entry between the insertion quotation marks will cause this error message *Edit 13* to be printed. Correct the error and resubmit the job.

***ERRØR (EDIT 14):
NØ END ØR CØNTINUATIØN WAS ENCØUNTERED.

No terminal punctuation mark or comma indicating additional modifications was found in the MØDIFY change listed above this message in the printout. Correct the error and resubmit the job.

***ERRØR (EDIT 15):
CHNGSTAK ØVERFLØW: DIVIDE THE CHANGE INTØ SMALLER PØRTIONS.

This message appears when the user has indicated a change that is very long. Divide the change into smaller changes and resubmit the job.

***ERRØR (EDIT 16):
THIS SENTENCE HAS ALREADY BEEN PASSED THRØUGH EDIT.

This message appears when the sentence number specified in a change refers to a sentence whose number is lower than the one currently in the computer. Either the change is out of order or the sentence number has been incorrectly specified. Correct the error and resubmit the job.

***ERRØR (EDIT 17):
THE DATA AND/ØR ØUTPUT FILES HAVE NØT BEEN SPECIFIED.

This message appears when the user has not specified the name of the file to be edited or the name of the file on which the edited material is to be written. Indicate the names of the files in the appropriate boxes on the program card and resubmit the job.

6.6. Additional Message

After certain errors have occured, an additional message will be printed out:

***NOTE: BEGINNING WITH THIS CHANGE, NØ FURTHER CHANGES WILL BE MADE; THE REMAINDER ØF THE CØRPUS WILL NØT BE MØDIFIED.

This message indicates that the current change contained an error, and that no additional changes will be made.

6.7. Programs Necessary to Execute EDIT

BEDRES, the input/output routines, GWEN, AUXGWEN, BETH, and EDIT are required to execute EDIT. The input/output routines are listed above.

Two programs must be executed in order to generate an index. The first, LISTER, goes through the text to be indexed and produces a list of all the different words which appear and an indication of the number of times each word appears. INDEX, the second program, takes that list and determines in which sentence each word appears and indicates by numbers the sentences in which each of the different words appear.

7.1. Semantic Units

LISTER and INDEX have the additional capability of handling semantic units; i.e., groups of up to *ten continuous words* with no intervening punctuation marks which have some meaning when taken together other than what could be expected from the meanings of the individual words. Examples of semantic units would include idiomatic expressions (e.g., *UP TIGHT*) and proper names (e.g., *HOUSE OF DAVID*). Semantic units are assigned labels as they are encountered in the text by LISTER and the occurrence of each semantic unit is reported in the final alphabetized list of words and index, under "semantic unit."

The first occurrence of a semantic unit must be specified by the user through the use of the "less than" symbol (<) and the "greater than" symbol (>) as in:

< UP TIGHT >

The programs will then note that the specified words represent a semantic unit, assign it a label (*SU001* through *SU999*) and search the remainder of the text for occurrences of that semantic unit. Only the first occurrence of each semantic unit in a given dataset should be bracketed. If more than 999 semantic units are encountered in a dataset LISTER will not perform properly. *No more than 999 semantic units can be put in a dataset.*

At the end of the LISTER printout, all semantic units and their labels will be written out:

SEMANTIC UNITS:

SU001: UP TIGHT

The alphabetized list of words output by INDEX will contain the labels for each semantic unit, the number of times each appeared and the numbers of the sentences in which each appears.

7.2. Conventions in LISTER and INDEX

LISTER and INDEX utilize only one convention not specified in the chapter on BEDRES and the Input/Output routines. *Words can have a maximum length of 16 characters.* Words longer than 16 characters are truncated.

7.3. Procedure for Executing LISTER

The following cards must be prepared for LISTER:

```
// EXEC BEDRES, TEXT='(textname)',WØRDNAM='(wordname)'
// SENTNAME='(sentname)'
//CØNTRØL DD*
**LISTER(TEXT:TEXT,CØL:1-80) (WØRDLIST:WØRDLST) (LISTING:decision).
/*
```

The parameter box labelled (*textname*) indicates the dataset to be analyzed. The parameter boxed labelled (*wordname*) and (*sentname*) on the *// EXEC* card contain the codewords for that particular dataset. As noted in the section concerning loading data onto the disk (see the COPY routines), each dataset has its own unique codewords. For example, a *// EXEC* card set up in the following manner:

```
// EXEC BEDRES, TEXT='JB01049M',WØRDNAM='YELLØW',SENTNAM='BLUE'
```

specified the text as *'JB01049M'* and indicates the codewords unique to that dataset, YELLØW and BLUE.

The LISTING box on the **LISTER* card is used to indicate whether a listing (*printout*) of the data generated by LISTER is desired. This printout lists the data and indicates the sentence number associated with each line of the text. The word *decision* is replaced by *YES* if the printout is desired and *NØ* if it is not.

7.3.1. Sample Program

```
/*ID PS=4545,NAME=BURNETT,CØDE=ANTHRØ
/*SETUP  UNIT=DISK,ID=DK0023
//  EXEC BEDRES, TEXT='JB01049M', WØRDNAM='YELLØW',
//  SENTNAM='BLUE'
//CØNTRØL DD *
**LISTER(TEXT:TEXT,CØL:1-80) (WØRDLIST:WØRDLST) (LISTING:YES).
/*
```

This sample program runs LISTER on the dataset labelled *JB01049M*, gives the appropriate codewords for this dataset, and asks that the data generated by LISTER be printed.

7.4. Default Options for LISTER

If the TEXT file name is not specified, the text is assumed to come from cards immediately following the program card. The TEXT records are assumed to be in the first 72 columns of each record. Not specifying the *WØRDLIST* parameter box will cause the list of different words to be sent to the printer. If no decision is indicated with regard to the listing of the data, no listing is produced.

7.5. Error Messages for LISTER

Three error messages may be printed out by LISTER.

```
***ERRØR (LISTER 1):
NØ CØRE AVAILABLE.
```

This message indicates that there is no core available for the list of different words appearing in the text. If the program is being run under MFT or MVT, request more core for the run. Otherwise, see a consultant at the computer installation.

***ERRØR (LISTER 2):
NØT ENØUGH CØRE AVAILABLE TØ HANDLE THIS CØRPUS.

This message indicates that the text to be handled is too big for the computer's memory. If the program is run under MFT or MVT, request more core and resubmit the job. Otherwise, break the text into smaller segments and submit each individually.

***ERRØR (LISTER 3):
UNRECØGNIZABLE PARAMETER FØUND.

This message indicates that the symbolic name for one of the parameter boxes was not "TEXT", "WØRDLIST", or "LISTING", or that the name was misspelled. Correct the error and resubmit the job.

7.6. Routines Required to Run LISTER

In BEDRES the input/output routines, GWEN and LISTER, must be included to execute LISTER.

8. INDEX

8.1. Procedure for executing INDEX

The following cards must be prepared for INDEX:

```
//  EXEC BEDRES, TEXT='(textname)',WØRDNAM='(wordname)',
//  SENTNAM='(sentname)'
//CØNTRØL DD *
**INDEX(TEXT:TEXT,CØL:1-80) (WØRDLIST:WØRDLST) (SENTLIST:SENTLST)
(LISTING:decision).
/*
```

The parameter boxed labelled *(textname)*, *(wordname)*, and *(sentname)* on the // *EXEC* card are handled in the same manner as they were in LISTER. Be sure that the codewords used are unique to the dataset to be analyzed. (See section on COPY routines.)

If the index (list of words appearing in the text and the sentence numbers in which the words appear) is to be printed, the box symbolically labelled LISTING should be coded so that *YES* replaced *decision*. If no listing is desired, *NO* replaces *decision* in the box.

8.1.1. Sample Program

```
/*ID PS=4545,NAME=BURNETT,CØDE=ANTHRØ
/*SETUP UNIT=DISK,ID=DK0023
//  EXEC BEDRES, TEXT='JB01049M',WØRDNAM='YELLOW',
//  SENTNAM='BLUE'
//CØNTRØL DD *
**INDEX(TEXT:TEXT,CØL:1-80) (WØRDLIST:WØRDLST) (SENTLIST:SENTLST)
(LISTING:YES).
/*
```

This sample program runs INDEX on the dataset *'JB01049M'*, gives the codewords unique to this dataset (yellow and blue) and asks for the data produced by INDEX to be printed.

8.2 Default Options for INDEX

The data are assumed to appear in the first 72 columns of each TEXT record. Unless otherwise specified, the index produced will not be printed. Unless otherwise specified, the list of sentence numbers will be written out on a dummy file, i.e., a non-existant file.

8.3 Cautionary Note

Since INDEX utilizes the output from LISTER the text operated on by both programs must be identical. For example, if a file has been *LISTERed* and then *EDITed,* it must be *reLISTERed* before the index can be completed.

If INDEX is not presented with the identical text as that which was used by LISTER to produce the WØRDLIST required by INDEX, an error [***ERRØR (INDEX 1), ***ERRØR (INDEX 2), ***ERRØR (INDEX 4), or ***ERRØR (INDEX 5)] will occur.

8.4. Error Messages for INDEX

***ERRØR (INDEX 1):
WØRD NØT IN THE LIST ØF DIFFEPENT WØRDS.

This message indicates that some word was encountered by INDEX but was not in the list of different words produced by LISTER. The probable cause of this error is the incorrect specification of the TEXT file. Correctly specify the file and resubmit the job.

***ERROR (INDEX 4):
THE TØTAL FREQUENCIES OF THE DIFFERENT WØRDS IS NØT THE SAME AS LISTER FØUND.

LISTER and INDEX both compute the total number of words in the text being examined. If the total arrived at by INDEX is not the same as

the total reached by LISTER, the above message is printed out. The source of the error is probably the same as indicated for ****ERRØR (LISTER 1)*. See that error above for the proper remediation.

***ERROR (INDEX 5):
DID NØT FIND THE SAME NUMBER ØF SENTENCES AS LISTER.

LISTER and INDEX both compute the total number of sentences in the text, and INDEX checks these totals to ascertain that they are the same. If they are not, the above message appears on the printout and execution terminates. For the sources of the error and the appropriate remediation, see ****ERRØR (INDEX 1)*.

***ERROR (INDEX 6):
THE TEXT FILE HAS NØT BEEN SPECIFIED.

This message indicates that no file name appeared in the parameter box symbolically named TEXT. Indicate the TEXT file name and resubmit the job.

8.5. Routines Required to Execute INDEX

In EEDRES the Input/Output routines, GWEN and INDEX, must be included in order to execute INDEX. The Input/Output routines are listed above. (See Chapter 4.)

9. CØNCØRD

The execution of the concordance program, CØNCØRD, for a given text requires that that text first have had an INDEX generated for it and stored for access. The generation of the index must be done by LISTER and INDEX and the output from these programs must be stored on some device, e.g., a tape or disk, to which CØNCØRD has access. Under certain circumstances, such as when the text to be processed is small or a permanent storage device is not available to the user, the index and concordance are all performed in one run.

9.1. Conventions used by CØNCØRD

As is the case in producing an index, CØNCØRD defines words as being 16 characters or less in length. Longer words are truncated to 16 characters.

9.2. Procedure for Executing CØNCØRD

The following cards must be prepared for CØNCØRD:

```
//  EXEC BEDRES,TEXT='(textname)', WØRDNAM='(wordname)',
//  SENTNAM='(sentname)'
//CØNTRØL DD *
**CØNCØRD(TEXT:TEXT,CØL:1-80) (WØRDLIST:WØRDLST) (SENTLIST:SENTLST)
(OUTPUT:PRINT,CØL:2-80,79) (LENGTH:pre,foll) (REQUEST:*).

    REQUEST CARD(S)-PLACED HERE

/*
```

The parameter boxes labelled *(textname)*, *(wordname)* and *(sentname)* on the // EXEC card are handled in the same manner as in LISTER and INDEX. Again, be sure that the codewords used are unique to the dataset to be analyzed.

The parameter box labelled LENGTH on the program control card refers to the number of sentences preceeding and following the sentences located in the concordance which are to be printed out as well. The number preceeding replaces "pre" in the box, and the number following replaces "foll".

Note the position of the request cards for CØNCØRD in the sequence above. These cards (described in section 9.4.) follow the program control card.

9.2.1. Sample Program

```
/*ID PS=4545 NAME=BURNETT,CØDE=ANTHRØ
/*SETUP  UNIT=DISK,ID=DK0023
//  EXEC BEDRES, TEXT='JB01049M', WØRDNAM='YELLØW'.
//  SENTNAM='BLUE'
//CØNTRØL DD *
**CØNCØRD(TEXT:TEXT,COL:1-80) (WØRDLIST:WØRDLST) (SENTLIST:SENTLST)
(OUTPUT:PRINT,CØL:2-80,79) (LENGTH:6,6) (REQUEST:*).
REQUEST:LØCATE ALL SENTENCES WITH THE WØRD ABLE.
/*
```

This sample program runs CØNCØRD on the dataset labelled 'JB01049M', gives the appropriate codewords for that dataset, and asks that six sentences preceding and six sentences following each concordance be printed. For this particular example, all sentences containing the word ABLE will be used in the concordance. It is possible, and indeed desirable, to use more than one REQUEST card in each job run.

9.3. Default Options for CØNCØRD

If the REQUEST file name is not specified, requests are assumed to come from cards immediately following the program card. If the ØUTPUT parameter box is left out, all output is sent to the printer. The default format prints the output in columns 6 through 125 of each line. The number of characters in each line defaults to 120. If LENGTH is

not specified, no sentences preceding or following the target sentence are printed.

9.4. Requests to CØNCØRD

Through the use of the filename specified in the REQUEST box, the user indicates to CØNCØRD where the words and phrases are that are to be "concordanced." These words and phrases take the form of logical requests--words bearing a relationship to one another that can be described in terms of the Boolean operators "and", "or", and "not". To the user, the logical requests take the form of simple declarative sentences which are then compiled into Boolean form by PATTI, the PATTern Interpreting routine. The Boolean form of the request is then printed out (so the user can ascertain that his logic was properly interpreted) and passed on to DIANA, which DIrectly ANAlyzes the Boolean code producing the concordance. The results of the direct analysis are passed to JØSHUA, a routine which, with the help of BETH and GWEN, locates and prints out the sentences filling the logical request specified by the user.

To interpret the Boolean form of the logical request, the user shpuld know that "&" means "and", "|" means "or", and "?" means "not". Sets of parentheses are used to group sets of words together.

In the event that an error is encountered in the processing of the request, PATTI or DIANA will print out a message indicating the error and the request is ignored. The most common error will be one of ambiguity. Such errors are caught by DIANA which then prints out a brief message indicating the nature of the ambiguity.

Example of logical requests include:

REQUEST:LØCATE THE SENTENCES WITH THE WØRD ABLE.

This sentence is then reported in Boolean code as

ABLE.

REQUEST:FIND THE ØCCURANCES ØF BØTH ABLE AND BAKER.

The Boolean code for this request is

ABLE&BAKER.

REQUEST:FIND THE SENTENCES NØT CØNTAINING ABLE.

This logical request is represented in Boolean code as

¬ABLE.

Concordances can also be generated for semantic units (see the description of semantic units in the Index chapter). The logical request should indicate the label of the semantic unit and not the words contained in it. Using the sample semantic unit presented in the preceding chapter, the logical request

REQUEST: FIND THE SENTENCES WITH SU001.

would result in the printing out of all the sentences containing "UP TIGHT".

9.5. Error Messages in CØNCØRD

In the execution of CØNCØRD error messages may be printed out by two routines, CØNCØRD and JØSHUA. The error messages are as follows:

***ERRØR (CØNCØRD 1):
UNRECØGNIZABLE PARAMETER FØUND.

This message indicates that a parameter box contained a symbolic name other than those specified in the calling sequence for CØNCØRD. Change the incorrect name and resubmit the job.

***ERRØR (CØNCØRD 2):
NØT ENØUGH CØRE AVAILABLE.

The core (memory) requirements exceed the core available. If the program is being executed under MVT or MFT, request more core and resubmit the job. Otherwise, see a consultant at the computer installation.

***ERRØR (CØNCØRD 3):
THE TEXT, SENTLIST, ØR WØRDLIST FILE NAMES HAVE NØT BEEN SPECIFIED.

At least one of the three file names listed in the message was not specified in the parameter box. Specify the correct files and resubmit the job.

***ERRØR (CØNCØRD 4):
THERE IS NØT ENØUGH CØRE FØR ALL THE DESIRED SENTENCES.
SEE PRØGRAM WRITE-UP.

More core (memory) is required for execution of the program than is available. If execution is conducted under MVT or MFT, request more core and resubmit the job. Under other circumstances, see a consultant at the computer installation.

***ERRØR (CØNCØRD 5):
THERE IS INSUFFICIENT SPACE AVAILABLE FØR THE SENTSTAK. SEE PRØGRAM WRITE-UP.

More core (memory) is required for execution of the program than is available. If execution is conducted under MVT or MFT, request more core and resubmit the job. Under other circumstances, see a consultant at the computer installation.

***ERRØR (JØSHUA)
DESIRED SENTENCE HAS BEEN PASSED.

Make the logical request immediately preceding the error message in the printout the first request in the job and resubmit.

9.6. Routines Required for the Execution of CØNCØRD

BEDRES, the input/output routines, GWEN, PATTI, BACKFILL, DIANA, JØSHUA, BETH, and CØNCØRD routines must be included to execute a concordance on a text.

9.7. Terminating Analysis of Datasets Through the Use of CØNCØRD

When the user of the BEDRES package feels that he has made full use of the capabilities of these programs on any particular dataset, *CØNCØRD should be given a dummy run to remove the codewords for that dataset.* As presently conceived, the BEDRES package standardized the analysis of any dataset by utilizing unique codewords for each block of data. These codewords are specified by the user when the data are first loaded onto the disk. However, a record of these codewords is also kept on the disk, creating a potential problem of filling the disk space with codewords for data which are no longer under analysis. To prevent filling the disk with these codewords, the following form of CØNCØRD should be run at the termination of data analysis of a given dataset.

```
//  EXEC BEDRES,TEXT='textname', WORDNAM='wordname'
//  SENTNAM='sentname', WORDISP='(OLD,DELETE)',SETDISP='(OLD,DELETE)'
//CONTROL DD *
**CØNCØRD(TEXT:TEXT,COL:1-80) (WORDLIST:WORDLST) (SENTLIST:SENTLST)
((OUTPUT:PRINT,COL:2-80,79) (LENGTH:0,0) (REQUEST:*)
REQUEST:LOCATE AA SENTENCES WITH THE WORD ABLE.
/*
```

The parameter box labelled *textname* should contain the name of the dataset which the use no longer wishes to analyze. The boxed labelled *wordname* and *sentname* should contain the appropriate codewords for that dataset. Should the user wish to analyze this dataset in the future, it will be necessary to reload the data onto the disk from the original cards or the back-up tape, and to specify the codewords again. Refer to the section under COPY entitled "Loading Data Onto the Disk".

APPENDIX A

TRANSFERRING FILES FROM DISK TO TAPE AND TAPE TO DISK

Purpose

Most events that are associated with humans are prone to error. Because of this fact it is useful to keep a spare or backup copy of the data files that are created with the BEDRES package. This can be expecially important when a large collection of data files has been created, and accidental loss of them could be very costly in human labor, time and money.

The BEDRES package is designed to work primarily with direct access storage devices. This method of data storage is usually one of the more expensive. To maintain a backup copy on one of these devices would, indeed, be very expensive. A much more economical method of long-term data storage is through the use of magnetic tapes.

Most computer manufactures supply with their machines a set of utility programs that serve the purpose of data management. For our purposes, a program that will create a backup copy of our direct access device on magnetic tape would prove to be very useful in the event that an accident did occur.

The Utility Program

Creating the Backup Copy

The example that follows is such a utility program provided by IBM for their operating system on the IBM System/360 computer. It involves execution of a program called IEHDASDR. The program copies a disk (in this example, DK0023) onto a standard label (an IBM convention) magnetic tape (Ø25985).

```
/*ID <accounting information>,IØREQ=10000
/*SETUP UNIT=TAPE,ID=Ø25985
/*SETUP UNIT=DISK,ID=DK0023
//DUMP EXEC PGM=IEHDASDR
//SYSPRINT DD SYSØUT=A
//DISK DD UNIT=DISK, VØL=SER=DK0023,DISP=ØLD
//TAPE DD UNIT=TAPE,VØL=SER=Ø25985,DISP=ØLD,DSN=DK0023
//SYSIN DD *
                    DUMP FRØMDD=DISK, TØDD=TAPE
/*
```

Restoring the Disk

This program then serves to move the contents of the disk onto magnetic tape for future use as a backup copy should it ever be needed (e.g., if the if the disk were broken).

If something does happen to our storage device then the following program will restore the disk to the same condition that it was when the backup copy was last created.

```
/*ID <accounting information>,IØREQ=10000
/*SETUP UNIT=TAPE,ID=Ø25985
/*SETUP UNIT=DISK,ID=DK0023
//RESTØRE EXEC PGM=IEHDASDR
//SYSPRINT DD SYSØUT=A
//DISK DD UNIT=DISK,VØL=SER=DK0023,DISP=ØLD
//TAPE DD UNIT=TAPE,VØL=SER=Ø25985,DISP=ØLD,DSN=DK0023
//SYSIN DD *
                    RESTØRE FRØMDD=TAPE TØDD=DISK
/*
```

Before restoring a direct access device to the condition that it was last in, it is important to take into consideration how much the datasets on the device have been altered since the dump program was last run, and how much was accidentally destroyed. Obviously any datasets that were changed or added since the last creation of the backup copy will need to be redone if the RESTØRE tape is used.

If a large amount of dataset manipulation has occurred since the creation of the backup tape and it becomes necessary to restore the disk, it might be useful to copy any of the newly created datasets onto some temporary storage device, restore the disk, then copy the newly created datasets back onto the disk, and finally replace the datasets that were destroyed, if any.

It would be greatly desirable to run the DUMP program whenever a significant number of changes have been made to the storage device such that rerunning these changes would be very costly in time and money. The ten statement DUMP program is easily run on the computer without any fear of destroying anything on the disk storage device.

APPENDIX B

SAMPLE OF USE OF BEDRES

Concatenation of data files

The following is a brief summary of one of our uses of the BEDRES programs.

After a sizable number of notes from the interviews had been keypunched from various interviewers, it was desirable to enter them onto the disk before they had a chance to become misplaced or confused with other notes. As the project came to a close, a huge amount of data had been collected and placed on the disk. The data had been entered on the disk in no particular order, other than by placing the data into a file coded by the name of the interviewer.

Many of the interviews involved follow-up interviews, so it was decided to rearrange the data in each interviewer's file so that it would be in chronological order. This gave an order to our huge collection of data that would facilitate ethnographic analysis and interpretations.

This process involved five steps:

1. Look up the date of each interview and the sentences numbers that it included. Do this for all of the data collected by each interviewer from a listing of that interviewer's files.
2. Arrange the interviews by informant and then arrange both interviews and fieldnotes in order by data.
3. Use the EDIT program to rearrange the interviews and fieldnotes by placing them in chronological order in a newly created file.

4. When finished with rearranging the data, use the LISTER program to obtain a listing of the newly ordered interviews and check for any errors.
5. Delete the old files if they are no longer needed for any other reason.

It should be noted that this process can be used for putting the files in any kind of order. Thus, if a case study was made in a particular area, or on certain people, the files might be ordered to reflect the nature of the study. After creation of the new files, INDEX and CØNCØRD can be used, if desired.

APPENDIX C

OBTAINING STORED FILENAMES AND CATALOGING

Obtaining Data Filenames Stored on Disk or Tape

After the BEDRES user has placed a considerable amount of data on either the disk and/or back-up tape, it may be useful to obtain a listing of all datasets found on these storage devices. This is accomplished by obtaining a list of all datasets (by filename) which the user has placed upon the disk or tape. Such a listing is made available by calling for a copy of the Volume Table of Contents (VTOC) found on the disk:

```
/*ID<accounting information>,Lines=20000,
/*IØREQ=5000,TIME=4
// SETUP,UNIT=DISK,ID=DK0023

// EXEC LISTDISK
//DD1 DD UNIT=2314,VOL=SER=DK0023,DISP=OLD
//SYSIN DD *
/*
```

Should the user wish to obtain the VTOC for the back-up tape, the data is simply switched from tape to disk (as outlined in Ap.B) and the above program is run.

Catolog Procedure for BEDRES

The speed, simplicity, and effectiveness of the BEDRES programs also depend upon the Catolog Procedure, since in its simplest form, the Catalog Procedure serves the function of setting up the BEDRES package on the computer in usable form *without* special instructions from the user. Should the catalog procedure be improperly altered or destroyed, the user of BEDRES would be required to find professional assistance in running the BEDRES programs. For this reason, the user of BEDRES is urged not to make changes in the catalog procedure listed below unless major revisions in the labeling or operation of the BEDRES programs are desired.

```
// PROC TEXT='TRIP',WORDISP='SHR',TEXTDSP='SHR',SETDISP='SHR',
// SENTNAM='SENTNAM',WORDNAM='WORDNAM'
// EXEC PGM=BEDRES,REGION=348K
//STEPLIB DD UNIT=DISK,VOL=SER=DK0023,DSN=USES.P2103.BEDRES,DISP=SHR
//PRINT  DD  SYSOUT=A,DCB=(LRECL=133,RECFM=FA)
//TEXT DD DSN=&TEXT,UNIT=DISK,VOL=SER=DK0023,DISP=&TEXTDSP,
// SPACE=(TRK,(4,4)),
// DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)
//WORDLST DD DSN=&WORDNAM,UNIT=DISK,VOL=SER=DK0023,DISP=&WORDISP,
// SPACE=(TRK,(4,4)),
// DCB=(LRECL=24,BLKSIZE=240,RECFM=FB)
//SENTLST DD DSN=&SENTNAM,UNIT=DISK,VOL=SER=DK0023,DISP=&SETDISP,
// SPACE=(TRK,(4,4)),
// DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)
//INPUT DD DDNAME=CONTROL
//CARD DD DDNAME=CARDS
//CHANGE DD DDNAME=CHANGES
```

GLOSSARY

This glossary is designed for the naive user and assumes no prior knowledge of computers in general and specifically, IBM System/360. Consequently, many of the definitions lack the technical rigor required by persons familiar with computing machinery.

Anchor Word
: In a MODIFY change specified for the EDIT program, any word which is not being modified but is used to indicate the position of the word(s) to be modified is an anchor word.

Assemble
: The procedure by which a source program is "translated" into code which can be executed. See source program and object module below.

Assembler Language
: The machine oriented language for IBM System/360 in which the BEDRES Package programs are written.

Auxiliary Storage Device
: Any device which can be used to store information but is not an integral part of the computer. Such devices must be mounted on the computer before they can be utilized.

BLKSIZE
: Records are often grouped together into blocks. In the Data Control Block (DCB), BLKSIZE indicates the number of bytes in such a block.

Boolean Operator
: In the production of concordances, the user's logical requests are translated into Boolean statements. The relationships between the words requested are represented by the operators and (&), or (|), and not (¬)--the Boolean operators.

Byte
: The unit of core (memory) required to represent one character.

Card Column (cc)
: An IBM card is divided into 80 vertical columns in which characters may be punched at a rate of one character per column. The character punched is usually printed out at the head of the column. Each column is called a card column and is identified by a number indicating its position on the card.

Carriage Control
When records are sent to the printer, i.e., when records are to be printed out, the first character of each record is used to indicate whether the printed lines are to be single, double, or triple spaced. The first character of each record is said to specify carriage control.

Concordance
A listing of all the sentences containing a word or constellation of words specified by the user. The Listing may include a certain number of sentences preceding and following each such sentence.

Control Card
Any card specifying to the computer's operating system which job steps (e.g., assemble, link edit, or execute) are to be performed, or defining a file to be used with a specified job step.

Convention
An arbitrary definition or procedure specified for some purpose. An example in the BEDRES Package is the definition of a sentence as any string of characters terminated by a period and the end of a line or a period and at least one blank space.

Core
The part of a computer in which information is stored. The information may be data or instructions.

Data Control Block (DCB)
The portion of a file defining control card which provides information regarding how data is stored on that file. The DCB specifies the length of each record (LRECL), the number of bytes in each block (BLKSIZE), and the format of the records (RECFM).

Data Definition (DD)
A type of control card defining a file for a given job step. The DD card describes the files to the operating system.

Data set
A set of records which is stored on a device such as a disk, tape, or deck of cards.

DD name
The name used on the program control card and the DD card to identify a data set used in the execution of a package program.

Default Option
If certain parameter inputs are not specified when a BEDRES package program is to be executed, the program assumes that particular files or options are to be used. These files or options are the default options. See the program chapters in this manual for a description of the default options for each program.

Direct Access Device
Any auxiliary storage device which allows the user to access information at any location on the device without having to start at the beginning.

Disk
A group of metalic disks with magnetic surfaces on which information can be stored.

DISP
On a data definition card, DISPosition indicates what is to be done with the data set defined in file represented by the DD card. The disposition indicates whether the data set is old, new or to be modified, and whether it should be retained (kept) or deleted.

DSNAME
The name of a data set. The DSNAME is specified on the data definition card.

Execute
Execute refers to the job step in which the task for which a given BEDRES Package program was designed is actually accomplished.

EXPLETIVE
Any of a set of words often used because of and to describe an unsuccessful run on the computer. These words include ****, ****, and ****.

Extended Binary Coded Decimal Interchange Code (EBCDIC)
The name of the pattern of punches used to represent characters on IBM cards for the IBM System/360.

File
A data set defined for use in some sob step.

Format
An indication of where on a record information can be read or written.

GIGØ
Garbage In Garbage Out. Refers to why nothing meaningful resulted from the processing of some data.

GØ
The execution job step.

Hard Copy
IBM cards or printout.

IBM Card
A specially cut and printed card made of heavy stock which can be punched on a keypunch and read by a card reader.

Index
A listing of the words appearing in a set of data and the numbers of the sentences in which each appears.

Input/Øutput (I/Ø)
Refers to the procedures for getting information into and out of the computer.

Job
A task to be performed by the computer.

Job Preparation
The preparation of a job is the collection and organization of control cards, program cards, and data as a prelude to submitting the job to the computer to be executed.

Job Step
Each step in the execution of a task is called a job step. CØ is an example of a job step.

Keypunch
A machine used to punch IBM cards. Keypunch may also refer to the act of punching cards.

Link Edit
The job step in which object modules are organized preparatory to their execution.

Load Module
A set of object modules that have been processed by the linkage editor and are ready to be executed.

Logical Record Length (LRECL)
The number of bytes (characters) in a record. LRECL is specified in the data control block on a data definition card.

Logical Request
A request to the CØNCØRD program indicating which word(s) is (are) to be "concordanced".

Memory
See Core.

MFT
Multiple Fixed Task (MFT) is an operating system under which several jobs can be processed simultaneously by the computer. Under MFT, core is divided into partitions of fixed size and the user requests a partition large enough to handle his job.

MVT

Multiple Variable Task (MVT) is an operating system under which several jobs can be processed simultaneously by the computer. Under MVT, the user specifies how much core his job requires and that much core is allocated for him.

NCAL

NCAL is a parameter for the link editing step indicating that programs may be mentioned in the object modules which do not appear in the job being link edited. If NCAL is not specified in a BEDRES Package job, the run will terminate prematurely.

Object Deck

An object module punched on cards.

Object Module

When a source program is assembled, the "translated" version of the program is referred to as an object module. The object module is the version of the program which is actually executed.

Operating System

The operating system is the programming which monitors and controls the computer's functioning. The user should see a consultant at his computer installation for information regarding the operating system under which his programs will operate.

Paragraph

BEDRES Package convention defines a paragraph as a string of characters the first five characters of which are blank.

Parameter Box

A parameter box is a pair of left and right parentheses containing information to be passed to one of the BEDRES package programs.

Printer

The output device which produces the printout. Occasionally, printer refers to the file which will be printed out; the file which is sent to the printer.

Printout

Printed output from the computer.

Program

A set of instructions which will be performed by the computer.

Program Control Card

A Program control card is one which indicates to BEDRES which of the package programs is to be executed. The card begins with two stars (**) in cc 1-2 and the first four characters of the program's name in cc 3-6. Following the program name are the parameter boxes described in the program's chapter in this manual. The program card is terminated by a period.

Punch

As a verb, punch refers to the act of keypunching of the production of punched cards by the computer, as in the case of the object deck. As a noun, it refers to the output device on the computer which produces cards or the keypunch.

Read

The procedure by which the computer examines records picking up the information they contain.

RECFM

Refers to RECord Format. RECFM is a parameter on the data definition card indicating the manner in which the file's records are organized. "F" indicates that the format is constant for all records, "B" indicates that records are blocked, and "A" indicates that the first character of each record is to be used as carriage control.

Record

A record is a string of contiguous characters, usually of some specified length. Line is a synonym for record.

Record Length

The number of bytes (characters) in a record.

Routine

Synonym for program.

Semantic Unit

A semantic unit is a group of up to ten contiguous words without intervening punctuation marks defined by the user as being of special interest.

Sentence

BEDRES Package convention defines a sentence as being a string of characters terminated by a period, question mark, or exclamation point and the end of the line or at least one blank.

SER

SERial number refers to the identification number of some device such as a disk.

Source Listing

The computer produces a printout indicating the location of a source program's instructions in core during the process of assembling the source program. This printout is called a source listing.

Source Program

A source program is a program before it has been assembled.

Stacking

Stacking is the running of several programs from the same job deck. It is accomplished by specifying the program cards for each program in the job deck in the order in which the programs are to be executed.

Stylistic Analysis

As defined by the BEDRES Package, stylistic analysis is the examination of some set of data using the sentence structure, phrase, and word choice considerations described in the chapter on Son of MOMSR.

Tape

A magnetic tape, not unlike those used on tape recorders, on which information can be stored.

TØLØNØ

A small town on Route 45 south of Champaign.

Truncate

Certain of the BEDRES Package programs define words as being no longer than 16 characters. Only the first 16 characters of words 17 characters and longer are used by such programs. Such words are said to be truncated to 16 characters. For example, the word ANTIDISESTABLISHMENTARIANISM would be truncated to ANTIDISESTABLISH.

UNIT

Any device such as a tape or disk.

Unit Record Device

Any device which handles "hard copy"; unit record devices include card readers, card punches, and printers.

VØL

VØLume is synonymous with device and refers to a tape, disk, etc.

Wheelbarrow

A unit of measure of data. A wheelbarrow corresponds roughly to "an awful lot".

Write

The procedure by which the computer puts records on some device.